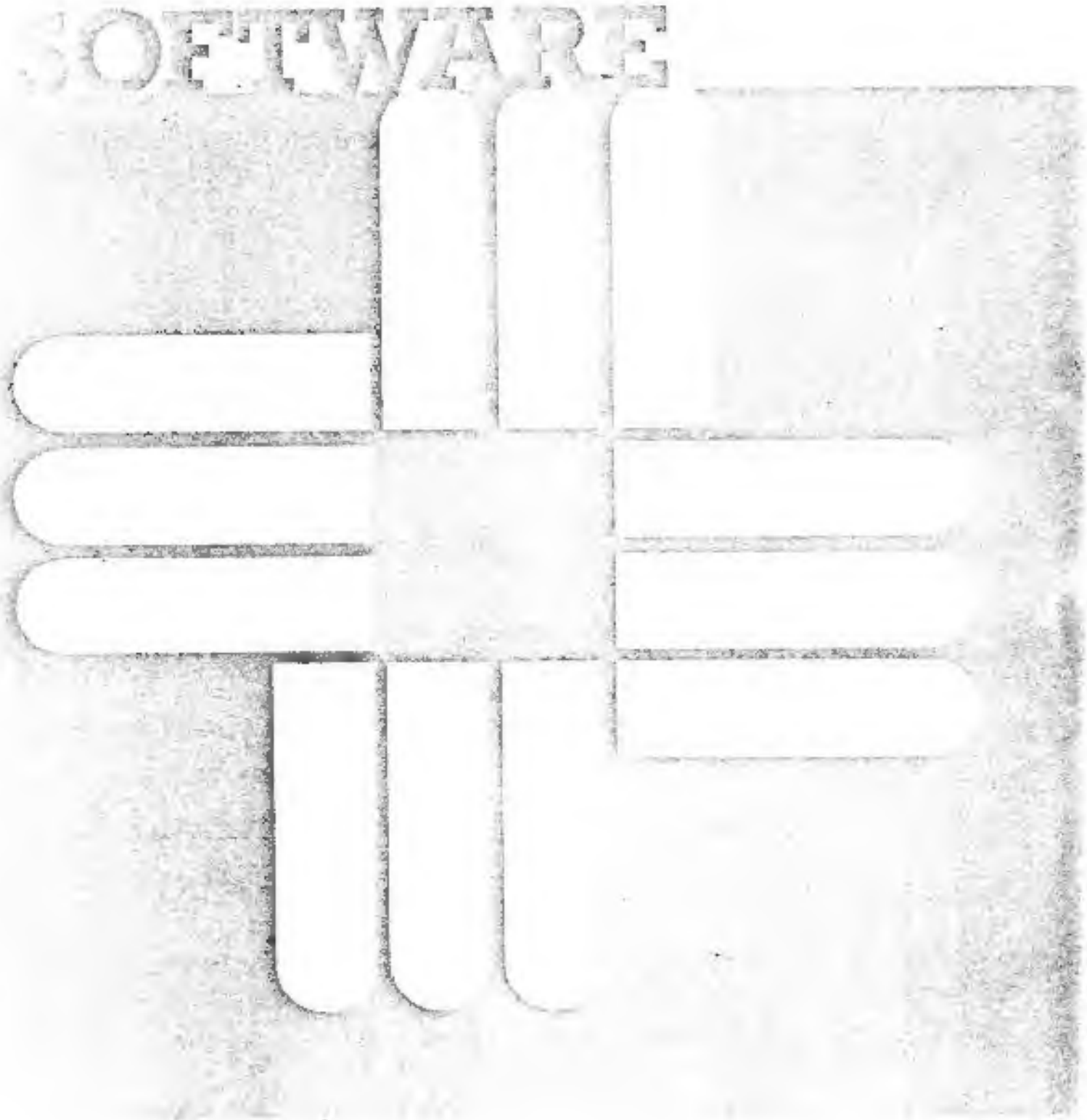
MODULAR COMPUTER SYSTEMS, INC.

Programmer's Reference Manual
MAX IV/MAX 32

Nonresident Job Control and Batch Facilities



REVISION INSTRUCTIONS

Revision H01, September 1985
MAX IV/MAX 32 PROGRAMMER'S REFERENCE MANUAL
NONRESIDENT JOB CONTROL AND BATCH FACILITIES
211-804002-H01

To update your manual, insert the enclosed pages according to the instructions:

Remove:	Insert:
Cover/Blank	Cover/Blank
iii/iv	iii/iv
v/vi	v/vi
vii/viii	vii/Blank
ix/Blank	ix/x
—	xi/Blank
3-11/3-12	3-11/3-12
3-13/3-14	3-13/3-14
3-15/3-16	3-15/3-16
3-19/3-20	3-19/3-20
3-23/3-24	3-23/3-24
3-25/3-26	3-25/3-26
3-27/3-28	3-27/3-28
3-29/3-30	3-29/3-30
3-33/3-34	3-33/3-34
3-37/3-38	3-37/3-38
3-39/3-40	3-39/3-40
3-41/3-42	3-41/3-42
3-45/3-46	3-45/3-46
3-49/3-50	3-49/3-50
3-59/3-60	3-59/3-60
3-61/3-62	3-61/3-62
3-65/3-66	3-65/3-66
3-67/3-68	3-67/3-68
3-71/3-72	3-71/3-72
---	3-72A/3-72B
3-79/3-80	3-79/3-80

3-81/3-82	3-81/3-82
3-85/3-86	3-85/3-86
---	3-86A/Blank
B-3/Blank	B-3/Blank
D-1/D-2	D-1/D-2
D-3/Blank	D-3/Blank
I-1/I-2	I-1/I-2
I-3/Blank	I-3/Blank

NOTE: Revised pages are marked with Revision Bars and the corresponding Revision number. When the manual is reissued, all outstanding revisions will be incorporated.

MANUAL HISTORY

Manual Order Number: 211-804002-H01

Title: MAX IV/MAX 32, Programmer's Reference Manual, Nonresident Job Control and Batch Facilities

Product Number: 600404-110H.2

Revision Level	Date Issued	Description
---	01/78	Initial issue.
---	06/78	Reissue (M.0 and D.0).
---	03/79	Reissue (N.0 and E.0).
---	07/80	Reissue (O.0 and F.0).
G00	10/82	Reissue (P.0 and G.0).
G01	07/83	Reissue (P.1 and G.1).
H00	09/84	Reissue (H.0). Changes include the addition of the $LFILE Directive, revision of the $FDx Directive and other enhancements. Manual title changed from MAX III/IV to MAX IV/MAX 32 Nonresident Job Control and Batch Facilities, Programmer's Reference Manual.
H01	09/85	Revision (H.2). FSP Directive was removed. The GIVE, TAKE and NUM Directives were added. File Manager files can now be expanded, but not contracted. The OPEN and CLOSE Directives were enhanced. Error message mnemonics CED and CCM were added, and EPD and CPD were changed to CEP and NUL.

Contents subject to change without notice.

PREFACE

Audience
This manual is directed to all users of the MAX IV and MAX 32 Operating Systems.

Subject
This manual describes the directives within the NONRESIDENT JOB CONTROL AND BATCH FACILITIES System Processor.

Supporting Products
JOB CONTROL operates under the MAX IV and MAX 32 Operating Systems.

Conventions Used In This Manual
Characters within brackets are optional. Within directives, lower case characters indicate that the value of the parameter must be supplied by the user.

Related Publications
The reader is referred to the following documents for additional information.

When ordering manuals, use the Manual Order Number listed below. The latest revision (REV) will be shipped.

Manual Order Number	Manual Title
213-804005-REV	MAX IV BASIC INPUT/OUTPUT SYSTEM System Guide Manual (SGM)
213-838005-REV	MAX 32 BASIC INPUT/OUTPUT SYSTEM System Guide Manual (SGM)
209-804001-REV	MAX IV FILE MANAGER File Management Manual (FMM)
213-838009-REV	MAX 32 FILE MANAGER System Guide Manual (SGM)
211-804011-REV	MAX IV AND MAX 32 TASK/OVERLAY CATALOGER Programmer's Reference Manual (PRM)
213-804001-REV	MAX IV GENERAL OPERATING SYSTEM System Guide Manual (SGM)
213-838001-REV	MAX 32 GENERAL OPERATING SYSTEM System Guide Manual (SGM)
211-804018-REV	MAX IV/MAX 32 FMSAVE Programmer's Reference Manual (PRM)
211-804019-REV	MA IV/MAX 32 FMLIST Programmer's Reference Manual (PRM)

Any reference in this manual to FORTRAN or COBOL pertains to FORTRAN Product Numbers 610203-000, 611203-000, 600202-000 and 600203-000 or COBOL Product Number 608250-029.

Users of MODCOMP FORTRAN 77 Product Number 620100-000 or MODCOMP COBOL 74 Product Number 620200-000 should refer to the Programmer's Reference Manual(s) for these languages and not to examples in this manual.

MODCOMP Product Training
MODCOMP's Education Services provides training courses on many hardware and software products at our Training Center in Ft. Lauderdale, Florida. On-site courses at customer sites can also be arranged. For more information about the courses offered by Education Services, contact the MODCOMP Training Registrar.

REVISION H.2 SUMMARY

The following enhancements were implemented in the H.2 revision of the MAX IV/MAX 32 NONRESIDENT JOB CONTROL AND BATCH FACILITIES System Processor. (The H.1 revision was never released.)

- Real File Manager files opened by multiple logical file names can be automatically or manually expanded but not contracted. CLOSE,,,ALL contracts a multiply-opened file when ACS is specified for the file.
- The Create Option (CR) was added to the OPEN Directive and the File Presence Option (FP) was added to the CREATE and DESTROY Directives to avoid aborts in Job Control procedures using these directives.
- The file name or volume name parameter is now optional in the alternate form of directive call for most File Manager directives.
- Increased the number of SYSPAGES in the JOBCT Installation Procedure to #09 to accommodate File Manager Services.
- Current service completion error codes were modified and new codes created for file EXPAND and CONTRACT Directives.
- Removed the FSP JM Directive.

REVISION H.0 SUMMARY

The following enhancements were implemented in the H.0 revision of the MAX IV/MAX 32 NONRESIDENT JOB CONTROL AND BATCH FACILITIES System Processor. This new revision operates under the I.0 Revision of the MAX IV Operating System or Rev A.0 of the MAX 32 Operating System, it will not work under MAX III.

- Procedure defaults - user is allowed to use % to default to an earlier parameter, example:

  ```
  $PROD NEWPROC,TY,NOLO,%1,NOMAP
  %3 defaults to whatever %1 value is.
  ```

- Handles all commands in upper or lower case.

- New construct to $IF statements uses THEN and ELSE as well as PRODUCE and semicolon (;).

  ```
  $IF %1=NO THEN $REW SI ELSE $ASS SI USL
  ```

- A double comma in $EXE statements designates a missing option.

- The new File Manager directive $LFILE has been added to list the contents of any File Manager directory, data or partition data file. This directive also supports data pattern searching.

- All File Manager directives now have an option (BS) to specify that the requested service be performed without using the default SYSGENed File Descriptor List (FDL).

- All File Manager directives can contain a filename (or volume name for volume related services) in place of a FDL specification (FDx). Other descriptors can be specified after this filename. This allows a File Manager call without previously defining an FDL.

- All File Manager directives can contain an imbedded volume name within the specified filename.

- The new form of the $FDx Directive ($FDx ?) lists the contents of a user's File Descriptor List.

- All File Manager errors list a text error message as well as the hexadecimal error code.

- Job Control has been modified to use REX services instead of going directly into map 0 to modify the THNA and BAT bits in the TCB.

- Job Control now allows command line arguments to be passed to a processor or a JM overlay. This is facilitated by placing them after an ! on the command line.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
OVERVIEW OF JOB CONTROL

1.1 WHAT IS JOB CONTROL?

All standard MODCOMP MAX IV/MAX 32 Operating Systems have a host batch processing task configured. This host batch processing task can be used to execute both MODCOMP system processors such as the Source Editor and user-written software.

The root overlay of the host batch processing task is called Job Control. Job Control is the program that is loaded when the system is initially started up, and whenever a user program aborts or exits. Job Control is also referred to as a system processor. When run from a terminal, Job Control always displays the dollar sign ($) prompt to indicate that it is ready to accept input.

1.2 LOGICAL FILES USED BY JOB CONTROL

Job Control processes all commands (direct vee) through logical files that can be assigned to different physical devices as needed.

The principal logical files used are:

Command Input (CI)	Directives are read from this logical file.
Computer to Operator (CO)	Messages to the operator's attention are written to this file.
Listing Output (LO)	Directives are written to this file.

Other logical files used by Job Control are:

UJM, JM	These logical files are used to store custom-written Job Control directives.
UJC, JC	These logical files are used to store Job Control procedures.
JW	This logical file is used as a work area by Job Control when performing procedures.
JBM	This logical file, if it exists, is searched on $EXE commands prior to searching the default logical file. JBM is only searched if the logical file is defaulted and JBM exists.

Use of custom-written directives and Job Control procedures is covered in Chapters 2, 3, and 4.

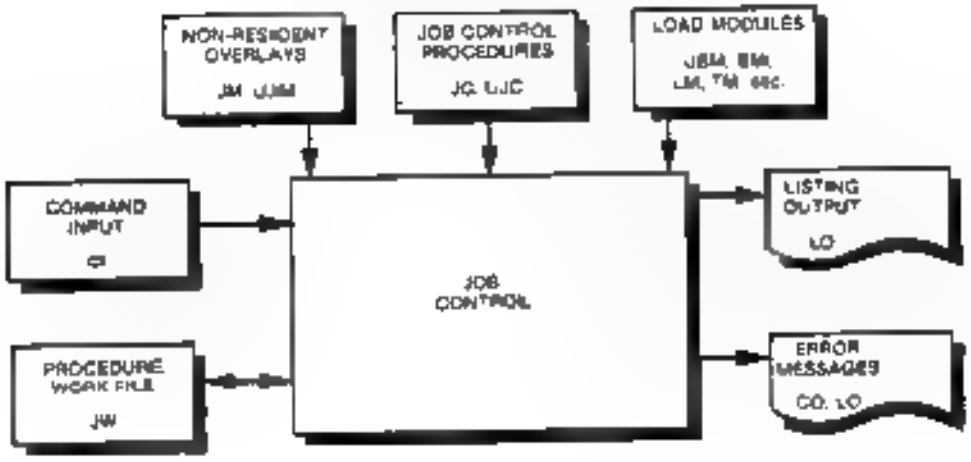


Figure 1-1. Logical Files Used by Job Control.

1.3 DIRECTIVE CATEGORIES

Job Control directives are commands that control the environment in which programs run. Directives fall into the following categories:

1. Information directives.
2. File control directives.
3. Program execution and task control directives.
4. File manager directives.
5. Procedure building directives.
6. Nonresident directives.

1.3.1 INFORMATION DIRECTIVES

Information directives send messages and identify comments. Information directives are $NOTE, $ACTION, $FORM, $ATTENTION and $NOP.

The $NOTE Directive prints a message containing the current time on the logical files CO and LO.

EXAMPLE

Logical files CI, CO, and LO are assigned to a terminal. After receiving the $ prompt from Job Control, the operator enters NOTE The time is. The last two lines are the response from Job Control.

```
$
NOTE The time is
$NOTE The time is 12:05:23 /B/B
$
```

/B/B in the above example represents the task name and the overlay name. Since Job Control is the root overlay of the host batch processing task, the task name and overlay name can be the same.

The $ACTION Directive is similar to the $NOTE Directive except that it causes the host batch processing task to enter a HOLD state. This feature is useful when the logical file CI is not assigned to a terminal, and operator verification is required before processing can continue. The operator can then choose to continue or to abort the job stream being processed.

If the logical file LO is assigned to a symbiont device such as an output spooler, the $FORM Directive halts printing to that device to enable a change of paper for a printer.

The $ATTENTION Directive sends a message to the operator's attention, and the $NOP Directive is a null statement that has no effect.

3.2 FILE CONTROL DIRECTIVES

File Control directives are $ASSIGN, $MOVE, $AVRECORD, $AVFILE, $HOME, $DEFAULT, $BKRECORD, $BKFILE, $REWIND, $WEOFILE and $POSITION. These directives are used to change logical file assignments, default assignments, and to position arbitrary logical files.

EXAMPLE

Logical file CI is assigned to a card reader. The following card deck writes a file-mark record after the tenth record of a magnetic tape mounted on tape drive MT1.

```
$ACTION Mount tape on drive MT1
$ASSIGN X MT1
$REWIND X
$AVRECORD X 10
$WEOFILE X
```

Three character abbreviations are accepted for all Job Control directives (except the $ENDFILE, $PROCEDURE, and $PRODEFAULT Directives that require four characters).

EXAMPLE

The following sequence of directives positions to the start of the file previous to file ABC on a magnetic tape containing a copy of a source library.

```
$ASS SI MT1
$POS ABC
$BKF SI 2
$AVF SI
```

1.3.3 PROGRAM EXECUTION AND TASK CONTROL DIRECTIVES

Program execution and task control directives are $JOB, $CJOB, $EOJ, $OPTION, $POPTION, $SET, $ALLOCATE, $DO, $LOCATE and $EXECUTE.

$JOB denotes the start of a new job stream. It resets all logical file assignments and options to their default values. It optionally closes all open File Manager files depending on how Job Control was assembled or cataloged. Refer to Appendix D for details. $EOJ marks the end of a job, resets all options, and writes a file-mark record to the LO device. $CJOB resets all options. If logical file CI is not assigned to a terminal device, any abort standardly causes the remaining directives from logical file CI to be ignored until a $JOB or $CJOB directive is encountered. An assembly option controls this feature. Refer to the $JO option in Appendix D.

$EXECUTE loads and executes programs and system processors. The remaining directives perform various functions concerning program execution. All directives are fully explained in Chapter 3.

EXAMPLE

This is an example of a complete job stream. It assembles and runs a program called TEST, the source of which is stored in a user source library (USL) file.

```
$JOB Assemble and run TEST
$ASS SI USL
$POS TEST
$REW BO
$EXE MSA
$WEO BO
$REW BO
$EXE MAIN,BO
$EOJ
```

1.3.4 FILE MANAGER DIRECTIVES

The File Manager Directives are $CLOSE, $CONTRACT, $CREATE, $DESTROY, $DISMOUNT, $ENDFILE, $EXPAND, $FILEDESCRIBE, $LABEL, $LFILE, $MOUNT, $OPEN, $REFILE, $RELABEL, and $FDx.

File Manager directives are only used if the File Manager System is configured. The MAX IV or MAX 32 File Manager File Management Manual should be read before using these directives.

1.3.5 PROCEDURE BUILDING DIRECTIVES

Procedure building directives are $COUNT, $IF, $IFNOT, $IFMISSING, $IFPRESENT, $EOF, $PROCEDURE, $PRODEFAULT, $GOTO, $ENDDO, $NOP, and $TAG.

Job Control procedures are sequences of Job Control directives stored on source files and usually cataloged in a source library. When a Job Control procedure is called, all the directives in it are processed. All procedures start with either a $PROCEDURE or a $PRODEFAULT Directive.

Any sequence of Job Control directives can be made into a procedure. The previous example could be made into a procedure as follows.

EXAMPLE

The following procedure file is cataloged through the Source Editor in the logical file USL under the name RUN.

```
$PROC RUN
$JOB
$ASS SI USL
$POS TEST
$REW 80
$EXE M5A
$WEO 80
$REW 80
$EXE MAIN,80
$EOJ
$ENDDO
```

This procedure can be executed using the $DO Directive.

```
$ASS JJC USL
$DO RUN
```

Logical files JJC and JC are used to store Job Control procedures. The $DO Directive is used to call a procedure. The procedure call line $DO RUN initiates exactly the same processing as entering all of the directives in the original example.

The $GOTO and $TAG Directives permit branching within procedures. The $COUNT Directive enables looping within procedures and the $IF, $IFNOT, $IFMISSING and $IFPRESENT Directives allow conditional statements to be performed.

Procedures can contain percent parameters that are replaced by parameters on the procedure call line ($DO) when the procedure is executed. Full details on how to build procedures and how to use percent parameters are given in Chapter 4.

1.3.6 NONRESIDENT DIRECTIVES

Nonresident directives are custom-written directives. Refer to chapters 2 and 4 for more information.

CHAPTER 2
SUMMARY OF DIRECTIVES

Job Control is the root task overlay of all host batch processing tasks. It is normally cataloged to start automatically when the operating system is activated, and restart whenever execution of a program terminates.

Job Control reads command lines (free-form command sentences known as directives) from the logical file CI.

The commands of the Job Control language are directives that begin with a dollar sign character $, and a directive name, followed by a list of required or optional parameters. Within a unit of work called a job, these directives are executed sequentially as they are received. A job is always started with a $JOB Directive and ended with a $EOJ (End-of-Job) Directive. In certain configurations of the MAX IV Operating System with multiple batch processing tasks being scheduled by an Input Spooler, individual jobs are not necessarily processed in the order that they are received from the job input device.

All directives should start with a dollar ('$') sign unless logical file CI is assigned to a terminal device. In this case, it is optional because a dollar sign prompt is output by Job Control to denote that it is running and ready to receive commands.

Parameters are seperated by any of the delimiters: comma (,), forward slash (/) or equal sign (=) or by one or more blanks.

Logical files JC and JM are normally used to store system procedures and nonresident directives. This convention makes logical files UJC and UJM available for user-specific applications.

If a directive is not recognized as a standard directive, the logical files UJM, JM, UJC and JC are searched in that order. If the directive is not found in any of them, an error message is generated.

Summaries of the directives processed by Job Control follow. These summaries are in a general functional order. A detailed description of each directive and its syntax is given in Chapter 3.

2.1 SUMMARY OF INFORMATION DIRECTIVES

$ACTION - Prints a message requesting system operator intervention; for example, mount volumes or change system status. Processing of the job does not proceed until the operator confirms that the requested action has been performed.

$NOTE - Prints a message to the attention of the system operator, but does not delay processing of the job.

$FORM - Prints a message instructing the operator to change forms on a spooled printing device. This message is not printed when received but is output to the spooler printer. When the spooled message is subsequently printed on a physical printer, the operator is notified. The spooled file is held and the spooler must be told when to resume printing.

$ATTENTION - Similar to the $FORM Directive except that the spooled file is not held.

$NOP - A "do nothing" null statement used mainly in $00 procedure prototypes. It is ignored by Job Control. However, it can be used to place comments in a job stream for documentation purposes. The directive following the $NOP Directive is processed.

2.2 SUMMARY OF FILE CONTROL DIRECTIVES

$ASSIGN - Relates logical files referenced in the next job stream directive or program to a physical device, real file, or another assigned logical file.

$DEFAULT - Relates the default assignment of logical files referenced in the next job stream directive or program to a physical device, real file or another assigned logical file. The $JOB directive initializes all logical file assignments to their default assignments.

$MOVE - Moves the assignment of an existing logical file and its sequential File Position Index to another logical file. This permits positional information in random access files to be saved for quick reference or for subsequent restoration.

$POSITION - Positions to a source file cataloged through the Source Editor. SI is the default logical file.

$REWIND
$AVRECORD
$AVFILE
$BKRECORD
$BKFILE
$HOME - These directives position logical files in preparation for using them. $REWIND positions every file to record zero of the assigned device or real file. The other operations move the File Position Index relative to the current position, either forward or backward.

$WEOF - Writes a file-mark record on a logical file. These marks can be used in subsequent positioning directives such as $AVFILE and $BKFILE.

2.3 SUMMARY OF PROGRAM EXECUTION AND TASK CONTROL DIRECTIVES

$OPTION - Changes the state (off = 0, or on = 1) for any system option bit in the task option word. These bits are interrogated by certain system services and processors in a manner similar to "sense switches".

$POPTION - S milar to the $OPTION D rect ve except that a d fferent option word is affected. The opt on word is the program opt on word which is unique to the host batch processing task. These option b ts are not used by the system, but can be free y used in the user's program.

$JOB - .n t.a.zes the .og cal file ass gnments, program opt ons, descr ptor ists, system options, and opt ons of the host batch processing task to the.r default va.ues as .nsta.led at system generation or as mod f.ed by the system operator at run-time. Opt onally c.oses all open F .e Manager f..es. Th.s direct ve .nit.alizes the batch task to begin a new job. Any job that does not start w.th this d rect.ve or a $CJOB D.rective is flushed from the input stream.

$CJOB - Sim.ar to the $JOB D.rective, but does not initia.ze .ogical fi.e assignments or descr.ptor l.sts or c.oses F..e Manager f..es that are a.ready open.

$ALLOCATE - A..ocates a certain amount of memory to the host batch processing task, so that the next job stream program has add t.ona. memory. If this directive .s not used, only enough memory .s allocated to f't the natural program s.ze. Each job step must specify add t.ona. memory .f it .s required.

$LOCATE - Relocates a program in the memory allocated to the host batch process.ng task .f under the MAX IV/MAX 32 Operating System. On.y uncata.oged programs processed by the Sequent al .oader or Link Loader can be so relocated. Batch processors or other cataloged programs are not normally cata.oged in a relocatable format, but can be relocated if cataloged as re.ocatab.e.

$EXECUTE - F.nds, .oads, and executes any named over.ay program or batch processor from a spec.f.ed logica. f.e. If the program .s found to be in cataloged quick .oad format, the MAX IV/MAX 32 standard loader is ca.led. If the program .s in any other object format, an appropr.ate loca. loader is ca..ed in as an overlay of Job Control, and it .oads or .ink-.oads the program. Contro. is then transferred to the .oaded program. Refer to Sect on 4.4, System option bits can a.so be changed by this command.

$SET - Sets sequentia. memory locat ons to a spec.fied va..e before the next program is .oaded.

$DO - Invokes a Job Contro. procedure that can consist of one or more statements and .nvolve one or more job steps. Parameters can be passed to the procedure.

$FOJ - Marks the end of a job, and not f.es the spoo.er that the pr.nting of spoo.ed printer f..es .og cal f.e LO for the .ob is complete. This statement a.so performs a F..e Manager CLOSE ALL funct on when the MAX IV/MAX 32 F.le Manager is configured.

2.4 SUMMARY OF FILE MANAGER DIRECTIVES

If the File Manager System is not installed at system generation, the following Job Control directives will not function.

The following File Manager directives initiate services that deal only with volumes of secondary storage and are known as "volume-related" directives:

$MOUNT	- Requests the mounting of a volume (disc or tape) on a transport (drive).
$LABEL	- Applies a standard label to a mounted volume.
$RELABEL	- Changes the label on a mounted volume.
$DISMOUNT	Prepares a volume for removal from a transport.

The remaining File Manager directives initiate services that manipulate the logical files of a task and/or the real files on a volume, and are known as "file-related" directives:

$CREATE	- Allocates space on a volume, and adds a new real file definition to a volume.
$REFILE	- Changes the file definition of an existing real file on a volume.
$ENDFILE	- Writes an "end-of-file" position of a real file in the file header label.
$EXPAND	- Increases the amount of space needed by a real file on a volume.
$FILEDESCRIBE	- Binds File Manager descriptors to a particular logical file name without invoking any particular File Manager service.
$OPEN	Makes a real file accessible to a program through a logical file reference.
$CLOSE	- Makes a real file no longer accessible to a program through its logical file reference.
$CONTRACT	- Decreases the amount of space used by a real file on a volume.
$DESTROY	- Removes a real file definition from a volume and deallocates its space.
$LFILE	- Lists the contents of any File Manager directory, data or partition data file.
$FDx	- Builds or lists file descriptor lists for the above services (refer to chapter 4 for details).

2.5 SUMMARY OF PROCEDURE BUILDING DIRECTIVES

All Job Control directives can appear in procedures. In addition, all the directives of other system processors can be used in a procedure. However, the following directives are appropriate only within procedures and are not strictly executed by Job Control. Instead, the direct expansion of procedures is executed.

$COUNT	- Facilitates looping and limit checking within a procedure.
$IF $IFNOT $IF MISSING $IFPRESENT	- These directives provide conditional testing of parameters and program conditions, thus making procedures adaptive to many needs.
$EOF	- Causes an file-mark record to be written on logical file JW.
$ENDDO	Causes a nested procedure to exit to the calling procedure or a primary procedure to return control to the initiating device.
$NOP	- Is used as a dummy target for conditional procedure branches, or to conditionally replace executable directives with a non-executable one.
$PROCEDURE	- Labels and defines the start of a procedure.
$PROCEFAULT	- Labels and defines the start of a procedure, and defines default values for percent parameters in the procedure.
$GOTO	- Provides the capability for jumping to a label within a procedure.
$TAG	- Defines a label within a procedure.

2.6 SUMMARY OF NONRESIDENT DIRECTIVES

Nonresident directives are relocatable overlays that can be written by the user. Each overlay processes one directive. The first six characters are significant in a MAX IV or MAX 32 environment. The overlays are stored in the load module logical files UJM and JM.

These overlays are loaded above Job Control in memory and have access to a buffer containing the command line. Upon completion they normally return to an address within Job Control. Alternatively, the REX service #12 (Exit) can be used.

Nonresident directives must be cataloged as relocatable. If they are to return to an address within Job Control, nonresident directives must also be cataloged as privileged. There is no maximum size for nonresident directive overlays in MAX IV or MAX 32.

In MAX 32 nonresident directives can be either 16-bit or 32-bit overlays. If a directive is privileged, it must be a 32-bit overlay. Load module files must not contain a mixture of 16-bit and 32-bit modules. All JM overlays supplied by MODCOMP are 32-bit overlays. If the user wants to maintain 16-bit overlays, they must be placed on UJM.

The entire command line is copied (all 80 bytes) to the command line extension of the TCB when the batch task has been catalogued with the CLE command. This extension is accessible to the JM through the task information REX.

The following values are passed to nonresident directive overlays in the registers specified.

R3,R4	6-character directive name in CAN-code
R6,R15	Address of command line
R7	Byte index. R6 and R7 are set so that the REX Collect (#35) and Get (#34) services can be used to get the next parameters.
R11	The ERROR return address
R13	A second error return address that prints up arrows beneath the last parameter accessed, provided R6 and R7 are intact
R14	The normal return address

CHAPTER 3
DIRECTIVES

HOW TO ENTER DIRECTIVES

Directives are commands used to initiate the functions within a system processor and to convey information needed by those functions. A directive consists of a directive name and parameters. The directive name initiates the system processor function. The parameters are arguments which convey needed information such as file names, key words, and numeric values. The following rules are standard when entering directives:

1. Only one directive can be entered per line.

2. In general, only the first three characters of directive names are significant. If more than three characters are entered, the additional characters must match the directive name. Procedure names have a maximum of 8 characters and must be entered in full.

3. Logical file names must be valid.

4. Numeric values are most commonly entered in decimal format. Hexadecimal values can be entered if preceded by the # character.

5. Standard delimiters to separate parameters within directives are:

   a. Space
   b. Comma (,)
   c. Forward slash (/
   d. Equal sign (=

6. One space is required between the directive name and the first parameter. Additional spaces are ignored. If using the space as a delimiter between parameters, one space is required and additional spaces are ignored. For example, entering any of the following variations of the same directive produces the same result.

```
CATALOG PROG1      SMITH
CATALOG PROG1,SMITH
CATALOG       PROG1 SMITH
CAT  PROG1, SMITH
```

7. The comma, forward slash, and equal sign can be used to denote use of the default for the parameter. For example,

   CAT,SMITH — denotes use of the default value for the first parameter and specifies SMITH as the value for the second parameter.

   CAT,,,SMITH — denotes use of the default values for parameters 1, 2, and 3 and specifies SMITH as the value for parameter 4.

$ACTION

WRITE A MESSAGE TO THE LOGICAL FILE CO AND ENTER A HOLD STATE

The $ACTION Directive writes a message for the operator's attention to the Computer-to-Operator (CO) logical file. If the remote flag is off for the batch task, Job Control enters the HOLD state, the Operator Communication (OC) task must be activated, and the /RESUME Directive must be entered to resume execution of the task. If the remote flag is on for the batch task, the remote terminal user need only enter the /RESUME Directive on the job entry terminal to resume execution of the task.

SYNTAX

$ACT[ION] [1 [text]]

[text] - Parameter 1 (optional) specifies the message directed to the operator's attention.

EXAMPLES

$ACTION SET THE WRITE-PROTECT SWITCHES FOR DISC

The message ACTION SET THE WRITE-PROTECT SWITCHES FOR DISC is written to the logical file CO.

$ACT MOUNT MAG-TAPE ON DRIVE 2

The message ACT MOUNT MAG-TAPE ON DRIVE 2 is written to the logical file CO.

ALLOCATE MEMORY FOR THE HOST TASK

The $ALLOCATE Directive allocates memory for the host task. This allocation remains in effect until Job Control is reloaded or until an overlay is loaded that deallocates the space (that is, DEALLOCATE ALL was specified when the overlay was cataloged in TOC).

The Link Loader (LINK option in the $EXECUTE Directive) and the Sequential Loader are loaded at the end of this memory area if specified on the next $EXECUTE Directive. If an $ALLOCATE Directive has not been specified and one of the loaders is needed, an $ALLOCATE ALL Directive is automatically performed before the loader is loaded. When the user issues an $ALLOCATE Directive explicitly, enough space must be included for the loaders if they are needed on the next $EXECUTE Directive. Refer to the $EXECUTE Directive for more information.

SYNTAX

$ALL[OCATE] [1 / words / ALL]

words
ALL - Parameter 1 (required) specifies the memory area allocated and must be one of the following:

words specifies the total amount of memory in words to be allocated to the addressing space of the host batch processing task. The number of words specified must be larger than the size of Job Control (4K words) and must be less than or equal to the instruction map size of the host task.

ALL specifies the entire instruction map of the host task is to be allocated. The maximum amount of memory that can be allocated is determined by the way the host task is cataloged as a task in TOC.

EXAMPLES

$ALLOCATE ALL	The entire instruction map of the host task is allocated.
$ALL 8275	The host task is allocated 8275 words of memory.
$ALL #2000	The host task is allocated #2000 words of memory.

$ASSIGN

LINK A LOGICAL FILE TO A LOGICAL FILE OR TO A DEVICE

The $ASSIGN Directive links a logical file(s) to a physical device(s) or to another logical file(s). A logical file assigned to itself (for example, $ASS LO LO) is assigned to its default assignment in the system. If there is no default, it becomes vacant.

A new file of the host batch processing task can be opened by the $ASSIGN Directive if that task has vacant file assignments in its File Assign Table (FAT). The logical file specified in parameter 1, if not already in the FAT, is opened if a vacancy exists. Such new files are not given a default assignment.

If the logical file specified in parameter 1 is assigned to a device, the File Position Index in the FAT is reset to zero. If it is assigned to another logical file, its File Position Index becomes that of the logical file specified in parameter 2.

Refer to the MAX IV BASIC INPUT/OUTPUT SYSTEM System Guide Manual for further information on logical files.

SYNTAX

	1	2
$ASS[IGN]	logfileone	logfiletwo devicename

logfileone - Parameter 1 (required) specifies the local logical file of the host batch processing task to be linked to a physical device or to another logical file.

logfiletwo
devicename - Parameter 2 (required) specifies one of the following:

logfiletwo specifies the logical file being linked to logfileone. A global file name can be specified in this parameter.

devicename specifies a physical device being linked to logfileone.

EXAMPLES

$ASSIGN LO=LP BI=DSA

Logical file LO is assigned to imaginary device LP. Logical file BI is assigned to physical disc device DSA.

$ASS LO LO

Logical file LO is assigned to logical file LO.

WRITE A MESSAGE TO THE LOGICAL FILE LO

The $ATTENTION directive writes a message on the listing device LO. If the listing device is not a spooled device, the message is printed on the logical file CO and Job Control processes the next statement.

If the listing device is a spooled device, a special carriage control byte (1) is prefixed to the output line. When this line is encountered by the printing part of the spooler, the spooler writes the line to its OUT file and continues printing. Refer to the MAX IV BASIC INPUT/OUTPUT SYSTEM System Guide Manual for more details on the spooler.

SYNTAX

$ATT[ENTION] [1 [text]]

[text] - Parameter 1 (optional) specifies a message of 76 characters maximum.

EXAMPLES

$ATTENTION MY JOB HAS FINISHED PRINTING

The message $ATTENTION MY JOB HAS FINISHED PRINTING is written to the logical file LO.

$ATT THIS IS MY JOB SAM

The message $ATT THIS IS MY JOB SAM is written to the logical file LO.

$AVA

DISPLAY AVAILABLE PORTS

The $AVA Direct ve 'a JM over ay ists al. the avai able ports (User ID=0) on a mult.-user systems. $AVA uses the PORTINFO REX serv ce to access the user ID information, wh ch is ma ntained by the MAGIC product.

SYNTAX

```
$AVA
```

EXAMPLE

$AVA

```
port  2
port  7
port  8
```

Multi-user ports 2,7 and 8 are not n use. The rest are activated.

ADVANCE A LOGICAL FILE A SPECIFIED NUMBER OF FILE-MARK RECORDS

The Advance File ($AVFILE) Directive advances a specified logical file (therefore, a physical device) a specified number of file-mark records. The File Position Index (FPI) in the File Assign Table (FAT) is incremented by 1 for each physical record encountered until the specified number of file-mark records are met. For a description of the response of a particular device to the $AVFILE Directive, refer to the MAX IV BASIC INPUT/OUTPUT SYSTEM System Guide Manual.

SYNTAX

	1	2
$AVF[ILE]	filename	[integer]

filename - Parameter 1 (required) specifies the name of the logical file to be advanced.

[integer] Parameter 2 (optional) specifies the number of file-mark records to be advanced. If not specified, the default value is 1.

EXAMPLES

$AVFILE 80 Advance logical file 80 for one file-mark record.

$AVF 81,2 Advance logical file 81 for two file-mark records.

$AVRECORD

ADVANCE A LOGICAL FILE A SPECIFIED NUMBER OF PHYSICAL RECORDS

The Advance Record $AVRECORD) Directive moves the specified logical file therefore, a physical device forward a specified number of physical records. The File Position Index (FPI) in the File Assign Table (FAT) is incremented by 1 for each physical record encountered. For a description of the response of a particular device to the $AVRECORD Directive, refer to the MAX IV BASIC INPUT/OUTPUT SYSTEM System Guide Manual.

If the end-of-medium position is reached before the specified number of physical records are encountered, the following message is displayed on the logical file CO which is usually assigned to the terminal:

END OF MEDIA

The specified logical file remains positioned beyond the last physical record until another file positioning directive is entered.

SYNTAX

	1	2
$AVR[ECORD]	filename	[integer]

filename - Parameter 1 (required) specifies the name of the logical file to be advanced.

[integer] - Parameter 2 (optional) specifies the number of physical records to be advanced. If not specified, the default value is 1.

EXAMPLES

$AVRECORD B0	Advance logical file B0 one physical record.
$AVR B1,2	Advance logical file B1 two physical records.

REVERSE A LOGICAL FILE A SPECIFIED NUMBER OF FILE-MARK RECORDS

The Back File ($BKFILE) Directive moves a specified logical file (therefore, a physical device, backward a specified number of file-mark records. If the particular device cannot perform such an operation, no movement occurs. If the beginning-of-medium position is reached before the specified number of file-mark records are passed, no additional movement of the physical device results.

The File Position Index (FPI) of the File Assign Table (FAT) is decremented by 1 for each file-mark record encountered. For a description of the response of a particular device to the $BKFILE Directive, refer to the MAX IV BASIC INPUT/OUTPUT SYSTEM System Guide Manual.

SYNTAX

	1	2
$BKF[ILE]	filename	[integer]

filename - Parameter 1 (required) specifies the name of the logical file to be reversed.

[integer] - Parameter 2 (optional) specifies the number of file-mark records to be reversed. If not specified, the default value is 1.

EXAMPLES

$BKFILE B1	Reverse logical file B1 one file-mark record.
$BKF B1,2	Reverse logical file B1 two file-mark records.

$BKRECORD

REVERSE A LOGICAL FILE A SPECIFIED NUMBER OF PHYSICAL RECORDS

The Back Record (SBKRECORD) Directive moves a specified logical file (therefore, a physical device) backward a specified number of physical records. If the particular device cannot perform such an operation, no movement occurs. If the beginning-of-medium position is reached before the specified number of physical records are passed, no additional movement of the physical device results.

The File Position Index (FPI) of the File Assign Table (FAT) is decremented by 1 for each physical record encountered. For a description of the response of a particular device to the $BKRECORD Directive, refer to the MAX IV BASIC INPUT/OUTPUT SYSTEM System Guide Manual.

SYNTAX

	1	2
$BKR[ECORD]	filename	[integer]

filename - Parameter 1 (required) specifies the name of the logical file to be reversed.

[integer] - Parameter 2 (optional) specifies the number of physical records to be reversed. If not specified, the default value is 1.

EXAMPLES

$BKRECORD BI	Reverse logical file BI one physical record.
$BKR BI,2	Reverse logical file BI two physical records.

PRINT A BANNER PAGE

The $BOX Directive (a JM overlay) prints a banner page at the beginning of a listing. The banner consists of the User ID and an ASCII string of up to eight characters. $BOX uses the PORTINFO REX service to access User ID information, which is maintained by the MAGIC product.

SYNTAX

$BOX char-str

char-str - Parameter 1 (optional) is an ASCII string of up to eight characters.

EXAMPLE

```
$ASS LO LP
$BOX TESTPROG
$POS TESTPROG
$EXE MSA,,NOSC,NOBO
$ASS LO LO
```

$BRO

BROADCAST A MESSAGE TO ALL TERMINALS

The $BRO D rect ve a JM overlay broadcasts a message preceeded by the Date/Time stamp f the DTS option is spec.fied in the SYSGEN) to al. term nals of a multi-batch system that have a nonzero User ID. This direct ve uses the PORTINFO REX service to access the User ID information, which is maintained by the MAGIC product.

SYNTAX

```
                    1
$BRO                message
```

message → Parameter 1 (required) is an ASCII string.

EXAMPLE

$BRO, There is a staff meeting at 2:00 p.m. today.

INDICATE THE BEGINNING OF A NEW JOB STREAM

The $CJOB Directive indicates the beginning of a new job stream. This directive performs the same functions as the $JOB Directive except that logical file assignments are NOT reset to their default assignments in the system and open File Manager files are NOT closed. If the currently executing program has aborted, File Manager descriptor lists are reset because Job Control is reloaded; otherwise, descriptor lists are not reset. Refer to Section 4.3.2.

SYNTAX

```
                    1
$CJO[B]           [text]
```

[text] – Parameter 1 (optional) specifies a user comment.

EXAMPLES

$CJOB

A new job stream is started.

$CJOB CONTINUE WITH THE NEW JOB STREAM

A new job stream is started. The comment is added to the directive line.

$CLOSE

CLOSE A FILE MANAGER FILE

The $CLOSE Directive closes a File Manager file and makes the file inaccessible to the user until the file is again opened.

SYNTAX

	1	2	3	4	5
$CLO[SE]	[logfile]	[FDx] [filename]	[RE[TAIN]] [descriptors]...	[AL[L]] [options]...	[BS[O]]

[logfile] - Parameter 1 (optional) specifies a 1- to 3-byte logical file name associated with the File Manager File that is to be closed. This parameter is required if the RETAIN parameter (3) is specified or the ALL option has not been specified. If retain is not specified, the logical filename is reassigned to its default assignment.

[FDx] - Parameter 2 (optional) FDx specifies the File Manager descriptor list FDx. Refer to the MAX IV or MAX 32 FILE MANAGER File Management Manual for a description of what descriptors are applicable to this service.

[RE[TAIN]] - Parameter 3 (optional) specifies that the File Control Block (FCB) associated with the logical file is to be kept. If not specified, the FCB is available for other files

[AL[L]] - Parameter 4 (optional) specifies that all open FCBs are closed and that all FCBs for this task are vacated.

[BS[O]] - Parameter 5 (optional) specifies that the File Manager service should be performed without using the SYSGEN defined default file descriptor list.

The alternate form of the directive:

[logfile] - Parameter 1 (optional) is the same as described above for the other format.

[filename] - Parameter 2 (optional) specifies the file name of the file to be closed. This form of the command assumes that this parameter is the file name, the FNA descriptor keyword should not be specified. The detailed description of the format of the filename is contained in the MAX IV or MAX 32 FILE MANAGER File Management Manual. This parameter can also contain a volume name specification in the form of: "!volnam/filename"

[descriptors]... - Parameter 3 (optional) specifies necessary File Manager descriptors instead of specifying the list with separate $FDx directives.

[options]... - Parameter 4 (optional) specifies any of the allowable options defined above for this service (Example: BSED). All options to be specified must appear after any descriptors specified in parameter 3 above.

EXAMPLES

$CLOSE SCA FDI RETAIN

The File Manager file associated with logical file SCA is closed using the file descriptors in the descriptor list FDI and the FCB associated with SCA is kept.

$CLOSE,F1,MYFILE VNA MYVOL BSD

The logical file F1 associated with the File Manager file MYFILE on volume MYVOL is closed without the use of the SYSGEN default file descriptor list.

$CLOSE,,,ALL

All open File Manager logical files associated with the batch task are closed. Automatic file contraction occurs if the ACS descriptor is specified for the file.

$COM

MODIFY THE COMMUNICATIONS PARAMETERS FOR A 1907 COMPATIBLE COMMUNICATIONS DEVICE

```
$COM  [device]  [BAU[DRATE] baud-rate                ]
      [*lfn]    [ECH[O] HAR[D]/SOF[T]/NO[ECHO]       ]
                [PARITY] ODD/EVE[N]/NON[E]           ]
                [STO[PBITS] 1/2                      ]
                [NUL[LS] number-of-nulls             ]
                [WAT[CHDOG] low-clock-ticks          ]
                [FRA[MESIZE] 5/6/7/8                 ]
                [SYN[CCHARACTER] hex-byte            ]
                [REC[ORDSIZE] number-of-bytes        ]
                [INF[LUENCE_LIMIT] number            ]
                [EXC[LUSIVEUSE] [taskname]           ]
                [MOD[E] CUR[RENTLOOP]/RS2[32]        ]
                [LIS[TCONTROL] ON/OFF                ]
                [RIN[G] BUS[Y]/NOT[BUSY]             ]
```

device
*lfn - Parameter 1 (optional) specifies either the device or the device assigned to by lfn.

The $COM Directive (a JM overlay) allows the user to display or modify the communications parameters of a 1907 compatible communications device. (A 1907 compatible communications device is a non-AFD device connected to one of the following controllers: 4804, 4806, 4807, 4808, 4809, 4858, 1907, 1907A, 1907B, or 1908.) Individual options of the command are described below:

$COM device BAU[DRATE] baudrate

This format allows the user to change the baud rate of the specified asynchronous 1907 device. The baud-rate parameter can take on one of the following baud rates: 75, 110, 134 (indicating a true baud rate of 134.5), 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, 9600, or 19.2 (indicating a baud rate of 19.2K). If the keyword EXTERN[AL] is used in place of the baud-rate parameter then clocking is performed by external modem control.

```
$COM device ECH[O]          HAR[D]
            SOF[T]
            NO[ECHO]
```

This format allows the user to change the presence and type of echo on the specified 1907 device. If the keyword HARD is given, then the hardware in the communications controller is used to perform the echo function. If the keyword SOFT is given, then the handler performs the echo function. If the keyword NOECHO is given, then no hardware or software echo is supplied.

```
$COM device PAR[ITY]        ODD
            EVE[N]
            NON[E]
```

This format allows the user to change the presence and type of parity to be generated/checked by the 1907 compatible controller. If the keyword ODD is given, then odd parity will be generated on output and odd parity will be checked on input. If the keyword

$COM device STO[PBITS] 1
2

This format allows the user to change the number of stop bits required for the specified asynchronous 1907 device.

$COM device NUL[LS] number-of-nulls

This format allows the user to change the number of NUL characters to be output to the specified 1907 device prior to each message.

$COM device WAI[CHDOG] low-clock-ticks

This format allows the user to change the maximum time in low clock ticks that an operation to the specified 1907 device is allowed to stay at the head of the controller queue. This format has no meaning if the device was SYSGENed UNTIMED.

$COM device FRA[MESZE] 5
6
7
8

This format allows the user to change the data character (frame) size in bits of the specified 1907 device.

$COM device SYN[CCHARACTER] hex-byte

This format allows the user to change the sync character for the specified synchronous 1907 device.

$COM device REC[ORDSIZE] number-of-bytes

This format allows the user to change the natural block size (bytes per record) of the specified 1907 device.

$COM device INF[LUENCELIMIT] number

This format allows the user to change the influence limit of the specified 1907 device. The influence limit is a number between 0 and 255 which specifies that only tasks having an influence limit that is less than or equal to the value specified will be permitted to use the specified 1907 device.

$COM device EXC[LUSIVEJSE] [taskname]

This format allows the user to change or remove the designated task which has (will have) permanent exclusive use of the specified 1907 device. The taskname parameter is a 1- to 6-character CAN-code name of a task which is required to have permanent exclusive use of the specified 1907 device. If the taskname parameter is omitted, then no task will have permanent exclusive use of the specified 1907 device.

$COM

$COM device [MOD[E] CUR[RENTLOOP]
RS2[32]

The format allows the user to change the mode of interface used for the specified 1907 device.

$COM device LIS[TCONTROL] ON
OFF

This format allows the user to change the Terminal Listing Control option of the specified asynchronous 1907 device.

$COM device RIN[G] BUS[Y]
NOT[BUSY]

This format allows the user to change the state of the Busy-Out modem signal of the specified 1907 device. If the keyword BUSY is given, then the Busy-Out signal will be set to active which causes a rotary device to ignore this channel (effectively excludes the channel from receiving a RING signal). If the keyword NOTBUSY is given, then the Busy-Out signal will be set to inactive which returns the channel to a "normal" state.

EXAMPLES

COM AAA BAU 9600	- Change the baud rate for AAA to 9600
COM AAA ECH NOECHO	- Change the echo for AAA to no echo
COM AAA PAR EVEN	- Change the parity for AAA to even parity
COM AAA STO 1	- Change the number of stop bits for AAA to 1
COM AAA NUL 3	- Change the number of nulls output to AAA to 3
COM AAA WAT 256	- Change the watchdog timer for AAA to 256 low clock ticks
COM AAA FRA 8	- Change the character size for AAA to 8 bits
COM SAA SYN #16	- Change the sync character for SAA to #16
COM AAA REC 72	- Change the record size of AAA to 72 bytes
COM AAA INF 100	- Change the influence limit for access to AAA to 100
COM AAA EXC	- Remove any permanent exclusive use from AAA

COM AAA MOD CURRENT_LOOP - Change the interface mode for AAA to current loop

COM AAA LIS ON - Allow AAA to use the Terminal Listing Control Option

COM AAA RIN NOTBUSY - Allow RING to occur on AAA

OUTPUT

```
$COM
                                    STOP          WATCH FRAM SYNC   REC INF
DEV BAUD ECHO PAR  MODE   LIST BITS NULLS  DOG   SIZE CHAR   SIZ LIM GJ.CHN
EXCLUS
A00 9600 NOEC NONE CURRE  ON   2        0     0    8  [NA]    0 255 18:001
A02 9600 NOEC NONE CURRE  ON   2        0     0    8  [NA]    0 255 18.003
A04 9600 NOEC NONE CURRE  ON   2        0     0    8  [NA]    0 255 18:005
A06 9600 NOEC NONE CURRE  ON   2        0     0    8  [NA]    0 255 18.007
A08 9600 NOEC NONE CURRE  ON   2        0     0    8  [NA]    0 255 18:009
A10 9600 NOEC NONE CURRE  ON   2        0     0    8  [NA]   40 255 18 011
A12 9600 NOEC NONE CURRE  ON   2        0     0    8  [NA]    0 255 18:013
A14 9600 NOEC NONE CURRE  ON   2        0     0    8  [NA]    0 255 18:015
A16 9600 NOEC NONE CURRE  ON   2        0     0    8  [NA]    0 255 18:017
A18 1200 NOEC NONE CURRE  ON   2        0     0    8  [NA]    0 255 18:019
A20 9600 NOEC NONE CURRE  ON   2        0     0    8  [NA]    0 255 18:021
A22 9600 NOEC NONE CURRE  ON   2        0     0    8  [NA]    0 255 18:023
$

$COM *TY
                                    STOP          WATCH FRAM SYNC   REC INF
DEV BAUD ECHO PAR  MODE   LIST BITS NULLS  DOG   SIZE CHAR   SIZ LIM GJ:CHN
EXCLUS
A16 9600 NOEC NONE CURRE  ON   2        0     0    8  [NA]    0 255 18:017


$COM *TY REC 27
                                    STOP          WATCH FRAM SYNC   REC INF
DEV BAUD ECHO PAR  MODE   LIST BITS NULLS  DOG   SIZE CHAR   SIZ LIM GJ:CHN
EXCLUS
A16 9600 NOEC NONE CURRE  ON   2        0     0    8  [NA]   27 255 18:017
```

$CONTRACT

DECREASE THE AMOUNT OF SPACE ALLOCATED FOR A FILE MANAGER FILE

The $CONTRACT Directive decreases the amount of space allocated to a File Manager file. The file must be on a randomly accessible media only. The file must have been previously opened before this directive is entered. Directory files, partition data files, and multiply-opened files cannot be contracted. The MAX IV File Manager does not allow the contraction of data files opened concurrently by more than one logical file.

SYNTAX

1	2	3	4
$CON[TRACT]	logfile	[FDx] [filename]	[MA[NUAL]] [descriptors]...	[BS[D]] [options]...

logfile - Parameter 1 (required) specifies the 1- to 3- character logical file name that was assigned to the File Manager file when it was opened.

[FDx] - Parameter 2 (optional) specifies the File Manager descriptor list FDx. Refer to the MAX IV or MAX 32 FILE MANAGER File Management Manual for a description of what descriptors are applicable for this service.

[MA[NUAL]] - Parameter 3 (optional) specifies that the amount of file space contracted is specified by the MCS descriptor in the file descriptor list FDx. An automatic contraction (ACS) is performed if ACS is specified and MCS is missing.

[BS[D]] - Parameter 4 (optional) specifies that the File Manager service should be performed without using the SYSGEN defined default file descriptor list.

The alternate form of the directive:

logfile - Parameter 1 (required) is the same as described above for the other format.

[filename] - Parameter 2 (optional) specifies the filename of the file to be contracted. This form of the command assumes that this parameter is the file name, the FNA descriptor keyword should not be specified. The detailed description of the file name format is contained in the MAX IV or MAX 32 FILE MANAGER File Management Manual. This parameter can also contain a volume name specification in the form of: "volnam/filename".

[descriptors]... - Parameter 3 (optional) specifies necessary File Manager descriptors instead of specifying the list with separate $FDx directives.

[options]... - Parameter 4 (optional) specifies any of the allowable options defined above for this service (Example: BS[D]). All options to be specified must appear after any descriptors specified in parameter 3 above.

EXAMPLE

$CONTRACT SCA FD1 MANUAL

The File Manager file associated with logical file SCA is contracted using the file descriptors in the descriptor list FD1. The amount of file space contracted is specified by the MCS descriptor.

$COUNT
$DO PROCEDURE ONLY

ESTABLISH A COUNTER WITHIN A PROCEDURE

The $COUNT Directive establishes a counter within a procedure. This directive can only be used within a procedure. The counter is established with an initial value. Each time the $COUNT directive is encountered (including the first time), the value is decremented by one. The directive can be used to skip records on the Command Input (CI) logical file when the counter becomes zero. In addition, control can be passed to a labeled statement by simulating a $GOTO Directive. An example of a procedure to list a source library file a specified number of times follows.

```
$PROC LISTN       Define a procedure named LISTN.
$ASS SI USL       Assign logical file SI to USL.
$TAG L1           Declare a label for the loop.
$EXE SED          Execute the Source Editor.
.POS %1           Position to file specified in parameter 1.
.LIS              List the file.
.EXI              Exit the Source Editor.
$COUNT %2,L1      Repeat the loop for the specified number in parameter 2.
```

This procedure can be called by the following command:

```
$LISTN,MYFILE,5
```

The above command would result in 5 listings of the file MYFILE. Full details on how to build procedures and how to use percent parameters are given in chapter 4.

SYNTAX

```
                    1          2
$COU[NT]        integer    [skiprecords]
                           [,labelname]
```

integer - Parameter 1 (required) specifies the initial value of the counter.

[skiprecords]
[,labelname] - Parameter 2 (optional) specifies one of the following:

skiprecords specifies the number of statements of the expanded procedure on the logical file JW to skip when the counter becomes zero.

,labelname specifies a label name defined elsewhere in the procedure by a $TAG Directive. Execution transfers to the label name whenever the counter is NOT zero.

EXAMPLES

$COUNT 25,1	A counter is established with an initial value of 25. When the counter becomes zero, one record is skipped.
$COU %1 ABC	A counter is established that can be passed an initial value through a percent parameter. Until the counter becomes zero, control is passed to the tag ABC. When the counter becomes zero, control is passed to the next directive.

$CPD

CHANGE PARTITION DEFINITION

The $CPD Directive is JM overlay changes the definition of a BIOS or File Manager disc partition that is either globally defined or locally assigned to the Batch or OC task.

SYNTAX

```
                    1     2       3       4      5
$CPD              ,fn  [strk]  [ntrks]  [rsl]  [geo]
```

fn - Parameter 1 (required) must be a logical file name assigned to the partition to be modified.

[strk] - Parameter 2 (optional) if present, is the new starting track of the partition. If omitted, starting track is not changed.

[ntrk] - Parameter 3 (optional) if present, is the new number of tracks of the partition. If omitted, number of tracks is not changed.

[rsl] - Parameter 4 (optional) if present, is the new record size limit of the partition. If omitted, record size limit is not changed.

[geo] - Parameter 5 (optional) if present, is the new geometry of the partition. If omitted, geometry is not changed.

NOTE: Validity of desired geometry should be verified by consulting the DISC DEVICE LOT description in the Data Structures manual.

EXAMPLE 1

```
/BATTSK.ASSBI BSB
  - or -
/,ASS BI BSB
  - or -
$ASS BI BSB
$CPD,BI,,,0,1
```

Change the partition assigned to BI to record size limit 0, geometry 1.

EXAMPLE 2

```
/OC/ASS X BSA
  - or -
//ASS X BSA
/CPD,X,,,0,,
```

Change the partition assigned to BI to record size limit 0, geometry.

NOTE: It is the user's responsibility to check for file contiguity of any FMGR partitions that are to be changed using the CPD directive.

CREATE A DIRECTORY ENTRY AND FILE MANAGEMENT INFORMATION FOR A NEW FILE

The $CREATE Directive creates a directory entry and file management information for a new file and records the entry and the information on the volume media at the appropriate hierarchical level.

SYNTAX

	1	2	3	4	5	6
$CRE[ATE]	[.logfile]	[FDx] [filename]	[RE[TAIN]] [descriptors]...	[IM[PATIENT]] [options]...	[BS[D]]	[FP]

[.logfile] - Parameter 1 (optional) specifies a 1- to 3-byte logical file name belonging to or being added to the host batch processing task and associated with a File Manager file. This parameter is required if the RETAIN parameter (3) is specified.

[FDx] - Parameter 2 (optional) specifies the File Manager descriptor list FDx. Refer to the MAX IV or MAX 32 FILE MANAGER File Management Manual for a description of what descriptors are applicable to this service.

[RE[TAIN]] - Parameter 3 (optional) specifies that the File Control Block (FCB) associated with the logical file is to be kept. If not specified, the FCB is available for other files.

[IM[PATIENT]] - Parameter 4 (optional) specifies that Job Control does not wait if the request cannot be satisfied immediately. If not specified, control is not returned until the file is created.

[BS[D]] - Parameter 5 (optional) specifies that the File Manager service should be performed without using the SYSGEN defined default file descriptor list.

[FP] - Parameter 6 (optional) specifies that Job Control should not report an error if the file already exists. If not specified, an error will be reported if the file was present before this directive was issued.

The alternate form of the directive:

[.logfile] - Parameter 1 (optional) is the same as described above for the other format.

[filename] - Parameter 2 (optional) specifies the filename of the file to be created. This form of the command assumes that this parameter is the file name, the FNA descriptor keyword should not be specified. The detailed description of the format of the filename is contained in the FILE MANAGER File Management Manual.

This parameter can also contain a volume name specification in the form of: "volnam/filename".

[descriptors]... - Parameter 3 (optional) can be used to specify necessary File Manager descriptors instead of specifying the list with separate $FDx directives.

[options]... - Parameter 4 (optional) can be used to specify any of the allowable options defined above for this service (Example: BS(D)). All options to be specified must appear after any descriptors specified in parameter 3 above.

EXAMPLES

$CREATE SCA FD1 RETAIN IMPATIENT

The directory entry and file management information for the File Manager file associated with logical file SCA is created using the file descriptors in the descriptor list FD1. The FCB is kept and Job Control does not wait for the volume to be mounted.

$CREATE,,FILEX

The data file FILEX is created using the SYSGEN default FDL and the internal File Manager defaults.

$CREATE,,FILEA FP

The data file FILEA is created using the SYSGEN default FDL and the internal File Manager defaults if it did not previously exist. If the file was already present, no error message is produced.

LINK THE DEFAULT ASSIGNMENT OF A LOGICAL FILE TO A LOGICAL FILE OR TO A DEVICE

The $DEFAULT Directive links the default assignment of a logical file(s) to a physical device(s) or to another logical file(s). This new default assignment is used by the $JOB Directive and when the user issues a logical file assignment to itself (Example: $ASS LO LO).

Every logical file has a default assignment associated with it when the JOB CONTROL task is initiated for each user. This directive allows the user to change this default assignment temporarily without need to change the resources of the user's specific JOB CONTROL task. This is only a temporary change.

If the specific JOB CONTROL task is reinitialized the original resources will be reloaded and the user must change the defaults again using this directive.

SYNTAX

	1	2
$DEF[AULT]	logfileone	logfiletwo devicename

logfileone - Parameter 1 (required) specifies the local logical file of the host batch processing task to be defaulted to a physical device or to another logical file.

logfiletwo
devicename - Parameter 2 (required) specifies one of the following:

logfiletwo specifies the logical file being defaulted to logfileone. A global file name can be specified in this parameter.

devicename specifies a physical device being defaulted to logfileone.

EXAMPLES

$DEF[AULT] LO=LP BI=DSA
Logical file LO is defaulted to imaginary device LP. Logical file BI is defaulted to physical disc device DSA.

$DEF[AULT] LO LO
Logical file LO is defaulted to logical file LO.

$DESTROY

REMOVE A SPECIFIED FILE MANAGER FILE FROM THE DIRECTORY

The $DESTROY Directive removes a specified File Manager file from the directory. Destroying a directory destroys all of its lower-level files.

SYNTAX

	1	2	3	4	5
$DES[TROY]	[logfile]	[FDx] [filename]	[IM[PATIENT]] [descriptors]...	[BS[D]] [options]...	[FP]

[logfile] - Parameter 1 (optional) specifies a 1- to 3-byte logical file name belonging to or being added to the host batch processing task and associated with a File Manager file.

[FDx] - Parameter 2 (optional) specifies the File Manager descriptor list FDx. Refer to the MAX IV or MAX 32 FILE MANAGER File Management Manual for a description of what descriptors are applicable to this service.

[IM[PATIENT]] - Parameter 3 (optional) specifies that Job Control does not wait if the request cannot be satisfied immediately. If not specified, control is not returned until the file is destroyed.

[BS[D]] - Parameter 4 (optional) specifies that the File Manager service should be performed without using the SYSGEN defined default file descriptor list.

[FP] - Parameter 5 (optional) specifies that Job Control should not report an error if the file does not exist. If not specified, an error will be reported although the file does not exist.

The alternate form of the directive:

[logfile] - Parameter 1 (optional) is the same as described above for the other format.

[filename] - Parameter 2 (optional) specifies the filename of the file to be destroyed. This form of the command assumes that this parameter is the file name, the FNA descriptor keyword should not be specified. The detailed description of the format of the filename is contained in the FILE MANAGER File Management Manual. This parameter can also contain a volume name specification in the form of: "!volname/filename".

[descriptors]... - Parameter 3 (optional) specifies necessary File Manager descriptors instead of specifying the list with separate $FDx directives.

[options]... Parameter 4 (optional) specifies any of the allowable options defined above for this service (Example: BS[D]). All options to be specified must appear after any descriptors specified in parameter 3 above.

EXAMPLE

$DESTROY SCA FDI IMPATIENT

The File Manager file associated with logical file SCA is removed from the directory using the file descriptors in the descriptor list FDI. Job Control does not wait if the file cannot be destroyed immediately.

$DESTROY,,FILEA FPRESENT

The file FILEA is destroyed if it currently exists. If the file is not present, no error is generated.

$DISMOUNT

REQUEST THE SYSTEM OPERATOR TO DISMOUNT A VOLUME

The $DISMOUNT Directive requests the system operator to dismount a volume. The volume cannot be dismounted if files on the volume are open.

SYNTAX

	1	2	3	4
$DIS[MOUNT]	[logfile]	[FDx] [volumename]	[BS[D]] [descriptors]...	[options]...

[logfile] - Parameter 1 (optional) specifies a 1- to 3-byte logical file name belonging to or being added to the host batch processing task.

[FDx] - Parameter 2 (optional) specifies the File Manager descriptor list FDx. Refer to the MAX IV or MAX 32 FILE MANAGER File Management Manual for a description of what descriptors are applicable for this service.

[BS[D]] - Parameter 3 optionally specifies that the File Manager service should be performed without using the SYSGEN defined default file descriptor list.

The alternate form of the directive:

[logfile] - Parameter 1 optional) is the same as described above for the other format.

[volumename] Parameter 2 (optional) specifies the volumename of the volume to be dismounted. This form of the command assumes that the parameter is the volume name, the VNA descriptor keyword should not be specified. The detailed description of the format of the volumename is contained in the FILE MANAGER File Management Manual.

[descriptors]... - Parameter 3 (optional) specifies necessary File Manager descriptors instead of specifying the list with separate $FDx directives.

[options]... - Parameter 4 (optional) specifies any of the allowable options defined above for this service (Example: BS[D]). All options to be specified must appear after any descriptors specified in parameter 3 above.

EXAMPLES

$DISMOUNT SCA FD1

The system operator is requested to dismount the volume identified by the file descriptors in the descriptor list FD1.

$DISMOUNT,,SYSVOL TNA T2

The system operator is requested to dismount the volume SYSVOL on transport T2.

$DO

ACTIVATE A PROCEDURE

The $DO Directive searches the logical file JC until the specified procedure name is found. The records following the procedure name (refer to the $PROCEDURE or $PRODEFAULT Directives) are written in uncompressed format to the logical file JW until either a file-mark record or another procedure is encountered. At that time, the Command Input (CI) logical file is assigned to the logical file JW.

Nested $DO Directives are allowed within procedures only if the logical file JW is assigned to a disc partition. If a nested $DO Directive is encountered, the nested procedure is expanded after the calling procedure on the logical file JW. After the execution of the nested procedure, the processing of the calling procedure resumes.

SYNTAX

	1	2
$[DO]	procedurename	[parameters]...

procedurename - Parameter 1 (required) specifies the name of the Job Control procedure.

[parameters]... - Parameter 2 (optional) specifies percent parameters that can be passed to the procedure.

EXAMPLES

$DO COMPILE CR,LP,PP,,NOMAP

The procedure named COMPILE is activated and the percent parameters CR, LP, PP, and NOMAP are passed to the procedure.

$COMPILE CR,LP

The procedure named COMPILE is activated and the percent parameters CR and LP are passed to the procedure. Note that the directive name DO is optional

TERMINATE A PROCEDURE

The $ENDDO Directive terminates a procedure. This directive can be used only within a procedure and only with a Job Control. When the $ENDDO Directive is encountered within a procedure, the procedure is immediately terminated. Control is returned to the calling procedure if the terminated procedure is nested. Otherwise, control is returned to the initiating device.

SYNTAX

```
                    1
$END[DO]        [text]
```

[text] - Parameter 1 (optional) specifies a user's comment.

EXAMPLES

$ENDDO
The procedure is terminated.

$END EXIT1
The procedure is terminated.

$ENDFILE

DEFINE THE END OF A FILE MANAGER FILE

The $ENDFILE Directive causes the data structure defining the END-OF-FILE condition to be recorded on the file's file header label. The directive cannot be used for a directory file.

This directive does not replace the WEOF REX service.

SYNTAX

	1	2	3	4
$ENDOF[ILE]	logfile	[FDx] [filename]	[BS[D]] [descriptors]...	[options]...

[logfile] - Parameter 1 (required) specifies the 1- to 3- character logical file name that was assigned to the File Manager file when it was opened.

[FDx] - Parameter 2 (optional) specifies the File Manager descriptor list FDx. Refer to the MAX IV or MAX 32 FILE MANAGER File Management Manual for a description of what descriptors are applicable to this service.

[BS[D]] - Parameter 3 (optional) specifies that the File Manager service should be performed without using the SYSGEN defined default file descriptor list.

The alternate form of the directive:

logfile - Parameter 1 (required) is the same as described above for the other format.

[filename] - Parameter 2 (optional) specifies the filename of the file that is to have its END-OF-FILE recorded. This form of the command assumes that this parameter is the file name, the FNA descriptor keyword should not be specified. The detailed description of the format of the filename is contained in the FILE MANAGER File Management Manual. This parameter can also contain a volume name specification in the form of: "!volnam/filename".

[descriptors]... - Parameter 3 (optional) specifies necessary File Manager descriptors instead of specifying the list with separate $FDx directives.

[options]... - Parameter 4 (optional) specifies any of the allowable options defined above for this service (Example: BS[D]). All options to be specified must appear after any descriptors specified in parameter 3 above.

EXAMPLE

$ENDFILE SCA FD1

The current FPL is recorded as the END-OF-FILE condition in the File Header label for the file associated with logical file SCA using the file descriptors in the descriptor list FD1.

$EOF
$DO PROCEDURE ONLY

WRITE A FILE-MARK RECORD ON LOGICAL FILE JW

The $EOF Directive writes a file-mark record on logical file JW. This directive can only be used within a procedure. When it is encountered during the expansion of the procedure, a file-mark record is written on logical file JW.

SYNTAX

```
                  1
$EOF            [text]
```

[text] - Parameter 1 (optional) specifies a user's comment.

EXAMPLES

```
$EOF
```

A file-mark record is written on logical file JW

```
$EOF THE END
```

A file-mark record is written on logical file JW.

WRITE A FILE-MARK RECORD ON THE LOGICAL FILE LO

The $EOJ Directive writes a file-mark record on the logical file LO. If the logical file LO is assigned to a line printer, a $$ line is written at the end-of-form position. The primary use of this directive is to write an end-of-file to a spooled printer. This end-of-file informs the spooler that the entire job is finished and that the spooler can now print another spooled job. When the MAX IV File Manager is used, the $EOJ Directive optionally closes all open files. Refer to the $WEOF Directive regarding file-mark writing for particular devices.

SYNTAX

```
$EOJ
```

EXAMPLES

$EOJ	A file-mark record is written to logical file LO.

$EXECUTE

LOAD A PROGRAM OR SYSTEM PROCESSOR INTO MEMORY AND EXECUTE IT

The $EXECUTE Directive loads a program or system processor into memory and executes it. The directive causes the code of Job Control to be overlaid by the code of the loaded program. The program is accessed from the logical file specified from logical file UBM if it exists, or from the default logical file that the batch task is loaded from. The MODCOMP naming convention for this logical file is UBM batch modules under MAX IV. If Job Control is running under MAX32, the program is also accessed from logical file VBM. For more information, refer to the MAX IV General Operating System, System Guide Manual, or the MAX 32 General Operating System, System Guide Manual listed in the preface.

The logical file or the default logical file can be a sequential access file or a special MAX IV quick-load directored file. Directored files are described fully in the MAX IV TASK OVERLAY CATALOGER Programmer's Reference Manual. If the file is a sequential access file, the word MAIN can be used instead of program-name. In this case, the logical file is not searched for a specific program; instead, the first program encountered is the one loaded. There is no special interpretation of MAIN if the logical file is a directored file.

Requests to load programs from a quick-load directored file can be satisfied first from UBM if it exists, or from the default logical file if the program was not found on the logical file. ASM or UBM load will exist. UBM is only searched when the logical file name is defaulted. If the user specifies a logical file UBM is not searched.

Desired settings of system options in the task option word can be included through the use of parameter 3. Refer to the $OPTION Directive for the valid option names. Job Control recognizes alternative names for options 10 through 16. The user can specify either 10 through 16 or 0 through 6. Prefixing the name with the 2-character string NO sets the bit to off. An option that is not called directly will not be changed.

LINK is a special Job Control option. If a program does not contain features specific to MAX IV (for example, counters), LINK can be specified. This option causes the system Link loader to load the program; otherwise, the sequential loader is used. If LINK is specified, the logical file in parameter 2 cannot be a directored file. In addition, the Link Loader does not rewind its input file.

The Link Loader processes programs from the logical files LB and UL in the following way: Programs loaded from the LB or UL logical files are not loaded until the end-of-file is encountered. Only the program specified is be loaded. A search to satisfy external references continues normally after the first module is loaded. Refer to Section 4.4.

User options are options that are passed to the executing program through registers. The registers passed to the executing program are set as follows:

R2 -	The address at which the executing program was loaded.
R3 -	The number of user options used from R8 through R15.
R6 -	The system option word.
R8 -	The user options. The number of user options used is specified in R4. If any register 8 through 15 is set to zero, a user option is not selected in that register. If a user option is specified, the first two bytes are passed in the register in ASCII. However, if "NO" precedes the user option, the option bit (Bit 0) is also set.
R15	

$EXECUTE

The $Execute Directive can contain comments or text to be passed to the program or processor being loaded. This is accomplished by placing an ! as part of the command. Job Control ceases scanning upon reading the exclamation point.

If the batch task was catalogued with the CLE option the entire command line (all 80 bytes) is copied to the command line extension. This data is then available to the program or processor through an option of the task information REX.

SYNTAX

$EXE[CUTE]	1 MAIN program-name	2 [logfile]	3 [LI[NK]] [[NO]system-options] [[NO]$user-options]	4 ['text]

MAIN program-name - Parameter 1 (required) specifies one of the following:

MAIN specifies that the first program encountered on the logical file is loaded.

program-name specifies the name of the program to be executed.

[logfile] - Parameter 2 (optional) specifies the name of the logical file from which the program is to be loaded. If not specified, the program is loaded from logical file UBM (if UBM exists) or VBM (if using MAX32) and the program is found or is loaded from the normal logical file for the host task's modules.

[LI[NK]]
[[NO]system-options]
[[NO]$user-options] - Parameter 3 (optional) specifies one or all of the following:

LINK causes the system Link Loader to load the program.

system-options specify desired settings of system options in the task option word (for example, MAP, LO or BO). The prefix NO can be used with these options and ,, (two commas) designate a missing option. For details, refer to the $OPTION Directive.

$user-options specify the name of a user option. The first two bytes of this name is passed to the executing program in a register. A maximum of eight user options can be passed. The prefix NO can be used with these options.

['text] - Parameter 4 (optional) allows entry of comments or command line arguments.

EXAMPLES

$EXE UPD
comments here or command line arguments.
The program UPD is executed.

$EXE PROG1,BL LINK,NOMAP,2,3,,NO4
The program PROG1 is executed from logical file BL. The Link Loader loads the program. The system options NOMAP, L2, L3 are set. The ,, designates a missing option. NO4 is set.

$EXE MAIN,BO..INK
The first program on logical file BO is executed through the Link Loader.

$EXE LIB,,NOLO
The system processor LIB is executed from the standard logical file. The system option NOLO is set.

$EXE PROG2,BL,L1,5C,$CARRY,NO$ALL,NO$3
The program PROG2 is executed from the logical file BI through the Link Loader. The system option 5C is set and is passed in R6. $CARRY, $ALL, and $3 are user options and are passed to PROG2 as follows:

```
R4          = 3     Three user options are used.
R8          = #4341  CA
R9          = #C14C  NOAL
R10         = #B320  NO3

R11 through 15 = 0
```

INCREASE THE SPACE ALLOCATION OF A FILE

The $EXPAND Directive increases the space allocation of a file on its volume. The file must have been previously opened before the execution of this directive. Directory files and partition data files cannot be expanded.

SYNTAX

	1	2	3	4
$EXP[AND]	.ogfile	[FDx] [filename]	[MA[NUAL]] [descriptors]...	[BS[D]] [options]...

.ogfile - Parameter 1 (required) specifies the 1- to 3- character logical file name that was assigned to the File Manager file when it was opened.

[FDx] - Parameter 2 (optional) specifies the File Manager descriptor list FDx. Refer to the MAX IV or MAX 32 FILE MANAGER File Management Manual for a description of what descriptors are applicable to this service.

[MA[NUAL]] - Parameter 3 (optional) specifies that the amount of file space expanded is specified by the MEI descriptor in the file descriptor list FDx. If not specified, an automatic expansion (AEI) is performed.

[BS[D]] - Parameter 4 optional specifies that the File Manager service should be performed without using the SYSGEN defined default file descriptor list.

The alternate form of the directive:

.ogfile - Parameter 1 (required) is the same as described above for the other format.

[filename] - Parameter 2 (optional) specifies the filename of the file to be expanded. This form of the command assumes that this parameter is the file name, the FNA descriptor keyword should not be specified. The detailed description of the format of the filename is contained in the FILE MANAGER File Management Manual. This parameter can also contain a volume name specification in the form of: "lvolnam/filename"

[descriptors]... - Parameter 3 (optional) specifies necessary File Manager descriptors instead of specifying the list with separate $FDx directives.

[options]... - Parameter 4 (optional, specifies any of the allowable options defined above for this service (Examples: BS[D]). All options to be specified must appear after any descriptors specified in parameter 3 above.

EXAMPLES

$EXPAND SCA FD1 MANUAL
The File Manager file associated with logical file SCA is expanded using the file descriptors in the descriptor list FD1. The MEI descriptor in FD1 is used.

$EXP SCA FD1
The File Manager file associated with logical file SCA is expanded using the file descriptors in the descriptor list FD1. The AEI descriptor in FD1 is used.

BATCH TASK'S FILE ASSIGNMENTS

The $FAT D rective (a JM overlay s a formatted l sting of the batch task's f.le assignments.

SYNTAX

$F AT [logica. f.lename]

[ogica. filename]

- Parameter 1 (optional), .f speci' ed, on.y the f.e assignment for that log cal f.e name is pr'nted. If omitted, the ent re .ist of f.e assignments is printed.

EXAMPLE

```
    $FAT  BI

NAM CUR DEF     FPI     DEV     TYPE RSL  GEO TNA VNA     FNA
BI  SCA SCA     0               0001 256  16  H10 SYSWRK /USR01S/SCA
```

$FDx

DEFINE OR LIST A FILE MANAGER DESCRIPTOR LIST

The $FDx Directive permits up to ten independent lists of File Manager descriptors to be constructed for use with File Manager service-invoking directives. The separate file descriptor lists are designated FD0 through FD9. This directive permits long lists of descriptors to be built with one or more statements. Once specified, the list can be referenced by name with service directives that invoke the desired File Manager service.

This directive is described in greater detail in Section 4.3 File Manager Services in this manual.

This directive can also be used to list a previously defined descriptor list.

SYNTAX

$FDx

$FDx (descriptor_specifications...) (?)
1 2

FDx - Parameter 1 (required) specifies which descriptor list is to be built or listed. Valid values for x are 0 through 9.

descriptor_specifications... - Parameters 2 to n (optional) contain the keywords and values to be used to define the descriptor list. Refer to Section 4.3.2 Building File Manager Descriptor Lists for a general description of the format for supplying descriptors. Refer to the FILE MANAGER File Management Manual for the descriptions of specific descriptors.

? - Parameter 2 (optional) contains the character "?" if the user wants to list a specific file descriptor list. Descriptors should not be specified on this form of the directive.

EXAMPLES

$FD0 TNA=T00 VNA MYVOL FNA=MYFILE
Descriptor list 0 has added to it the descriptor codes and values for transport name, volume name, and filename.

$FD1 INI FNA MYDIR/TMPFILE
Descriptor list 1 is initialized and then built with the descriptor code and value for filename.

$FD0 ?
The descriptor list 0 is listed.

ASSOCIATE THE PARAMETERS OF A FILE MANAGER FILE WITH A LOGICAL FILE

The $FILEDESCRIBE Directive associates the parameters of a File Manager file with a logical file. This directive selects the specified logical file and initializes a File Control Block (FCB) to the default descriptors if the file has not been previously in use by the task. If the FCB identified by the logical file name is already in use, the descriptors supplied with this directive are bound to the FCB. Once descriptors are bound to the FCB, they do not need to exist in the Job Control descriptor list FDx. Rather, the descriptors occupy primary storage in the Task Control Block of the host batch processing task.

SYNTAX

	1	2	3	4
$FIL[EDESCRIBE]	logfile	[FDx] [filename]	NO[DEFAULT]] [descriptors]...	[BS[O]] [options]...

logfile - Parameter 1 (required) specifies the 1- to 5-character logical file name that will be linked to a new FCB created for this service or the logical filename that has already been linked to the requested file/volume by another File Manager service.

[FDx] - Parameter 2 (optional) specifies the File Manager descriptor list FDx. Refer to the FILE MANAGER File Management Manual for a description of file descriptors.

[NO[DEFAULT]] - Parameter 3 (optional) specifies that the system default descriptor list (specified at SYSGEN time) is not used. If this parameter is missing, the system default descriptor list is used before the supplied descriptor list. System descriptors can be overridden by re-specifying them in the supplied descriptor list because the last specified descriptor takes precedence.

[BS[O]] - Parameter 4 (optional) specifies that the File Manager service should be performed without using the SYSGEN defined default file descriptor list.

The alternate form of the directive

logfile - Parameter 1 (required) is the same as described above for the other format.

[filename] - Parameter 2 (optional) specifies the filename of the file to be described. This form of the command assumes that this parameter is the file name, the FNA descriptor keyword should not be specified. The detailed description of the format of the filename is contained in the FILE MANAGER File Management Manual. This parameter can also contain a volume name specification in the form of: "[volnam/filename".

$FILEDESCRIBE

[descriptors]... - Parameter 3 (optional) specifies necessary File Manager descriptors instead of specifying the list with separate $FDx directives.

[options]... - Parameter 4 (optional) specifies any of the allowable options defined above for this service (Example: BS[0]). All options to be specified must appear after any descriptors specified in parameter 3 above.

EXAMPLES

```
$FD0 VNA=SYSVOL TNA=M0
$FD0 FNA=SOURCE/GEORGE/PROG6
$FIL SCA FD0
```

The File Manager file SOURCE/GEORGE/PROG6 is associated with logical file SCA using the file descriptors in the descriptor list FD0.

```
$FIL,SCA,'SYSVOL/SOURCE/GEORGE/PROG6 TNA=M0
```

This example produces the same results as the previous example.

DISPLAY FILE MANAGER OFFLINE FILES

The $FOL Directive to JMV overlay displays any File Manager files that are offline for the batch task.

SYNTAX

```
$FOL
```

EXAMPLE

```
$FOL
                    NO FILE MANAGER FILES OFF-LINE

$FOL
                    LFN = SCA IS OFFLINE,     TASK = BAT020
                    LFN = BI  IS OFFLINE,     TASK = BAT020
```

$FORM

SKIP A LINE AND WRITE A MESSAGE TO LOGICAL FILE LO

The $FORM Directive skips a line and writes a message on the listing device LO. If the listing device is not a spooled device, the message is printed on the CO file and the host task enters a HOLD state. The user must then activate the Operator Communication (OC) task and issue a /RESUME directive in order to resume execution of the task.

If the listing device is a spooled device, a special carriage control byte (?) is prefixed to the output line. When this line is encountered by the printing part of the spooler, the spooler writes the line to its OUT file and enters a HOLD state. This directive can be used for change of forms. Refer to the MAX IV BASIC INPUT/OUTPUT SYSTEM System Guide Manual for more information on the spooler.

SYNTAX

```
                 ]
$FOR[M]          [text]
```

[text] - Parameter 1 (optional) specifies a user's comment up to 76 characters in length.

EXAMPLES

$FORM MAGTAPES ARE SWITCHED NOW

A line is skipped and the message $FORM MAGTAPES ARE SWITCHED NOW is printed on the listing device LO.

$FOR SWITCH PAPER ON LINE PRINTER

A line is skipped and the message $FOR SWITCH PAPER ON LINE PRINTER is printed on the listing device LO.

GIVE UP EXCLUSIVE USE OF DEVICE ASSIGNED TO LOGICAL FILE

The $GIVE Directive permits the user to remove previously imposed exclusive use conditions from one or more devices. The user must refer to the devices by the names of the logical file(s) to which the device(s) are assigned.

SYNTAX

$GIV[E] 1 logfilname...

logfilname — Parameter 1 (required) must be the name of a logical file of the executing batch task or of the system's Global File Assign Table.

EXAMPLES

$GIVE DO

Give up exclusive use of device assigned to the DO file.

$GIVE X

Give up exclusive use of device to which logical filename, X, is assigned.

$GOTO

TRANSFER CONTROL WITHIN A PROCEDURE

The $GOTO Directive transfers control within a procedure to a subsequent or previous label. If the label is not found, the procedure aborts.

SYNTAX

$GOT[O] labelname

labelname - Parameter 1 (required) specifies a label name defined elsewhere in the procedure by the $TAG Directive.

EXAMPLES

$GOTO A1A

Control is transferred to the label A1A.

NORMALIZE DEVICE

The $HOME Directive normalizes or homes a specified logical file (therefore, a physical device). If the particular device cannot perform such an operation, no movement occurs.

SYNTAX

	1	2
$HOM[E]	logfile	[logfile]...

logfile - Parameter 1 (required) specifies the name of the logical file to be homed.

[logfile] - Parameter 2 (optional) specifies additional logical files to be homed.

EXAMPLE

$HOME B1,B0,L0

The physical devices associated with logical files B1, B0 and L0 are normalized.

$IFMISSING
$IFPRESENT
$DO PROCEDURE ONLY

TEST FOR THE PRESENCE OF PERCENT PARAMETERS

The $IFMISSING and $IFPRESENT Directives are used only within procedures. The directive name must adjoin the initial dollar sign. These directives are an exception to the general rule that spaces can appear between the dollar mark and the directive name. These directives are used to test for the presence of certain specified percent parameters in a procedure call statement ($DO statement). The specified parameter is tested for being present or not present and the result of the test is true or false depending on the keyword $IFM or $IFP. Some action (as specified by the second parameter) is performed as a result of the test. This action is the processing of one of the following directives. These directives replace the original test directive when the procedure is expanded. These two directives are:

- $AVR C1,n in which n represents the number of records to skip as specified in parameter 2.
- $NOP if the PRODUCE option is specified in parameter 2. The directive specified by the PRODUCE option is processed immediately after the $NOP Directive.

The exact substitutions for percent parameters depend on the exact contents of both the parameter list on the procedure call line ($DO) and the default parameter list on the $PRODEFAULT definition line. The following cases can be distinguished:

1. The parameter is present (non-space) on the call line. References to the parameter within the procedure are replaced with up to eight characters of the parameter specified on the call line.

   EXAMPLE

   ```
   $DO SQUEEZE BSL
           The percent parameter %1 within the procedure is replaced by the
           value BSL.
   ```

2. The parameter is omitted from the call line but a corresponding non-space default parameter exists on the $PRODEFAULT definition line and the first character of the default parameter is not a percent sign (see items 4 and 5 below). References to the parameter within the procedure are replaced with up to eight characters of the default parameter specified on the $PRODEFAULT line.

   EXAMPLE

   ```
   $DO SQUEEZE
   $PRODEFAULT SQUEEZE BSL
           The percent parameter %1 within the procedure is replaced by the
           value BSL.
   ```

3. The parameter is omitted on the call line and the corresponding default parameter is an embedded omitted item on the $PRODEFAULT definition line. References to the parameter within the procedure are deleted from the expansion of the procedure on the logical file JW.

EXAMPLE

$DO SQUEEZE BSL
$PRODEFAULT SQUEEZE BSL,,ALL

The percent parameter %2 within the procedure is omitted from the call line and is an embedded omitted item on the $PRODEFAULT definition line. All references to %2 within the procedure are deleted from the expansion of the procedure on the logical file JW.

4. The parameter is an embedded omitted item on the call line and the corresponding default parameter is an embedded omitted item or is a non-space string starting with a percent sign on the $PRODEFAULT definition line. References to the parameter are deleted from the expansion of the procedure on the logical file JW.

EXAMPLE

$DO SQUEEZE BSL,,ALL
$PRODEFAULT SQUEEZE BSL,,ALL or $PROD SQUEEZE BSL,%YES,ALL

The percent parameter %2 within the procedure is an embedded omitted item on the call line and on the $PRODEFAULT definition line or begins with a percent sign on the $PRODEFAULT definition line. In this case, references to the parameter within the procedure are deleted from the expansion of the procedure on the logical file JW.

5. The parameter is a trailing omitted item on the call line and the corresponding default parameter is also a trailing omitted item or is a non-space string starting with a percent sign on the $PRODEFAULT definition line. References to the parameter within the procedure are not modified and so appear as the original %character in the expanded procedure on the logical file JW

EXAMPLE

$DO SQUEEZE BSL,YES
$PRODEFAULT SQUEEZE BSL,YES or $PROD SQUEEZE BSL,YES,%ALL

The percent parameter %3 within the procedure is a trailing omitted item on the call line and is a trailing omitted item or a non-space string starting with a percent sign on the $PRODEFAULT line. References to the percent parameter %3 within the procedure appear as %3 within the expanded procedure on the logical file JW.

$IFMISSING
$IFPRESENT
$DO PROCEDURE ONLY

SYNTAX

	1	2	
$IFM[ISSING] $IFP[RESENT]	%character	skip-records P[RODUCE] commandone THEN commandone	[;commandtwo] [ELSE commandtwo]

%character - Parameter 1 (required) specifies the percent parameter to be tested. The percent parameter is replaced by the first eight characters of the corresponding parameter on the procedure call line ($DO).

skip-records
P[RODUCE] commandone [;commandtwo]
THEN commandone [ELSE commandtwo] - Parameter 2 (required) specifies one of the following:

skip-records specifies a positive integer that indicates the number of expanded directive statements (records on logical file JW) that are skipped if the test is true. Actually, the test directive itself is replaced during expansion with the executable directive $AVR CI,skip-records if the test is true, or with the executable directive $NOP if false.

PRODUCE or THEN is a keyword that causes the string commandone to be executed next if the test yields a true result. The optional second string commandtwo (commandtwo must be preceded by a semicolon (;) or the string ELSE) is executed next if the test yields a false result. If commandtwo is not present, the directive following the test directive is executed next (that is, the $NOP directive replaces the test directive in the expanded procedure on the logical file JW).

commandone is any valid statement, command, or data record that can be executed in the context of being expanded on the logical file JW if the test directive yields a true result. It must not contain a semi-colon.

commandtwo is any valid statement that can be executed if the test directive yields a false result.

EXAMPLES

$IF M %1,9

If the test for percent parameter %1 as missing is true, the directive $AVR CI,9 replaces the test directive in the expanded procedure on logical file JW. Therfore, the logical file CI is advanced 9 records. The directive at that position in the procedure is executed.

$IF PRESENT %8,PRODUCE,$ASSIGN BI=SI BO=SO
or
$IF PRESENT %8 THEN $ASSIGN BI=SI BO=SO

If the test for percent parameter %8 as present is true, the test directive is replaced by the $NOP Directive and the $ASSIGN BI=SI BO=SO Directive is executed next.

$IF MISSING %5 P $EXEC TOC ;$EXEC LIBUP
or
$IF MISSING %5 THEN $EXEC TOC ELSE $EXEC LIBUP

If the test for percent parameter %5 as missing is true, the test directive is replaced by the $NOP Directive and the $EXEC TOC Directive is executed next. If the test of %5 is false, the test directive is replaced by the $NOP Directive and the $EXEC LIBUP is executed next.

$IF
$IFNOT
$DO PROCEDURE ONLY

TEST RELATIONSHIPS BETWEEN PARAMETERS

The $IF and $IFNOT Directives test relationships between some value and a parameter on the procedure call line ($DO) or the corresponding default parameter on the $PRODEFAULT definition line. In addition, these directives can test relationships between two parameters on the procedure call line. The relational test can check for equality of a parameter to some string value (up to eight characters) or it can test the numerical value of a parameter.

The two expressions to be compared must be separated from each other as shown in the syntax. In particular, comparison for equality occurs when a single simple delimiter (blank, comma, slash, or equal sign) separates the expression. Comparison for less than or greater than occurs when the appropriate operator (< or >) appears between the expressions with one significant delimiter on both sides of the operator.

Expression comparison occurs when up to eight characters of the referenced parameter on the call line are expanded and compared with up to eight characters of the other expression. This is a character-by-character string comparison.

SYNTAX

	1	2	
$IF $IFNOT	expression expression expression,expression expression=expression expression/expression expression > expression expression,>,expression expression < expression expression,<,expression	skip-records P[RODUCE]	commandone [;commandtwo] THEN commandone [ELSE commandtwo]

expression – Parameter 1 (required) specifies an expression in one of the following forms:

The expression can be a percent parameter of the form %p that references the corresponding parameter on a call line or to corresponding default parameter. The first eight characters of the referenced parameter replace the percent parameter when the test directive is evaluated at procedure expansion.

The expression can be a string of up to eight characters to be compared with the replaced characters of a referenced percent parameter.

sk p-records - Parameter 2 (requ red) specifies one of the fo.ow.ng:
P[RODUCE] commandone [;commandtwo]
THEN commandone [EL SE commandtwo]

skip-records spec.fies a positive integer that ndicates the number of expanded direct ve statements (records on logical fi.e JW) that are sk.pped if the test is true. Actually, the test direct ve itself is rep.aced during expansion w.th the executable directive $AVR CI,skip-records if the test is true, or w.th the null directive $NOP if false.

PRODUCE or THEN is a keyword that causes the string commandone to be executed next if the test yields a true resu.t. The optional second string commandtwo (commandtwo must be preceded by a semicolon (;) or the string ELSE) is executed next if the test yields a false result. If commandtwo is not present, the directive following the test d rect ve is executed next (that .s, the $NOP directive repaces the test directive .n the expanded procedure on the logical file JW).

commandone is any valid statement, command, or data record that can be executed .n the context of being expanded on the logical f.le JW if the test directive yields a true result. It must not contain a semi-colon.

commandtwo .s any va.id statement that can be executed if the test directive yields a false result.

EXAMPLES

$IF %5 NOLO 6

If percent parameter %5 equals NOLO, the directive $AVR CI,6 replaces the test directive .n the expanded procedure on logical file JW. Therfore, the logical file CI is advanced 6 records. The direct ve at that position in the procedure is executed.

$IFNOT %4 %5 PRODUCE $REWIND LO ;$WEOF LO
or
$IFNOT %4 %5 THEN $REWIND LO ELSE $WEOF LO

If the test for percent parameter %4 not equa. to %5 s true, the test d rect.ve .s replaced by the $NOP Directive and the $REWIND LO Directive is executed next. If the test is false, the test directive .s repaced by the $NOP Directive and the $WEOF LO .s executed next.

$JOB

INDICATE THE BEGINNING OF A NEW JOB STREAM

The $JOB Directive indicates the beginning of a new job stream. It causes Job Control to do the following:

1. Reset all the logical file assignments to their DEFAULT assignments in the system. The Command Input (CI) logical file is the only logical file whose assignment is not affected by this directive.
2. Clear all non-permanent logical file assignments (if any) created by a previous user.
3. Initialize the task's system option word to its default state and reset the GO-NOGO flag in the task's Job State word.
4. Reset the $LOCATE address to the start of the background task (normally zero). Refer to the $LOCATE Directive.
5. In MAX IV, close all open File Manager files and reset all descriptor lists as the default option.

The entire directive $JOB... is written on the logical file CO unless the logical files CO and CI are both assigned to the same device. A $JOB or $CJOB Directive must be entered after a job stream has been aborted; otherwise, all following Job Control directives are ignored.

SYNTAX

$JOB [,1 [text]]

[text] - Parameter 1 (optional) specifies a user's comment.

EXAMPLES

$JOB COMPILE/EXECUTE

A new job stream starts with the message $JOB COMPILE/EXECUTE being written to the logical file CO.

$JOB

A new job stream starts with the message $JOB being written to the logical file CO.

APPLY A VOLUME LABEL TO AN UNLABELED VOLUME

The $LABEL Directive applies a volume label to an unlabeled volume. Mounting of the required volume can be accomplished within this directive or can be performed with a separate $MOUNT Directive.

SYNTAX

	1	2	3	4	5
$LAB[EL]	[logfile]	[FDx] [volumename]	[RETAIN] [descriptors]...	[IM[PATIENT]] [options]...	[BS[O]]

[logfile] - Parameter 1 (optional) specifies a 1- to 3-byte logical file name belonging to or being added to the host batch processing task and associated with a File Manager volume. This parameter is required if the RETAIN parameter (3) is specified.

[FDx] - Parameter 2 (optional) specifies the File Manager descriptor list FDx. Refer to the MAX IV or MAX 32 FILE MANAGER File Management Manual for a description of applicable descriptors for this service.

[RE[TAIN]] - Parameter 3 (optional) specifies that the File Control Block (FCB) associated with the logical file is to be kept. If not specified, the FCB is available for other files.

[IM[PATIENT]] - Parameter 4 (optional) specifies that Job Control does not wait if the request cannot be satisfied immediately. If not specified, control is not returned until the request is satisfied.

[BS[O]] - Parameter 5 (optional) specifies that the File Manager service should be performed without using the SYSGEN defined default file descriptor list.

The alternate form of the directive:

[logfile] - Parameter 1 (optional) is the same as described above for the other format.

[volumename] - Parameter 2 (optional) specifies the volumename of the volume to be labeled. This form of the command assumes that the parameter is the volume name, the VNA descriptor keyword should not be specified. The detailed description of the format of the volumename is contained in the FILE MANAGER File Management Manual.

[descriptors]... - Parameter 3 (optional) can be used to specify necessary File Manager descriptors instead of specifying the list with separate $FDx directives.

$LABEL

[options]... - Parameter 4 (optional) can be used to specify any of the allowable options defined above for this service. Example: BS[OD]. All options to be specified must appear after any descriptors specified in parameter 3 above.

EXAMPLE

```
$FD1 VZN=DATAB VZO=J.STEWART VDA=6/12/83 VNA=DATAB
$FD1 VOW=A.BAKER TNA=M1
$LABEL,FD1
```

The volume under volume name DATAB, with volume owner A.BAKER, and on transport M1 is labeled with volume name DATAB, volume owner J.STEWART, and volume date 6/12/83.

IDENTIFY AND LIST THE CONTENTS OF A FILE MANAGER FILE

The $LFILE Directive identifies the type of data contained within the requested File Manager file and produces a list of the file appropriate to its type. This directive supports the File Manager file types of directory, data and partition data files. This directive can identify the following types of file contents:

- File Manager directories
- Source Editor directories (SED)
- Compressed ASCII
- Uncompressed ASCII
- SED Save
- SED work file
- Library update directory (LIB & LIB32)
- Library update save (LIB & LIB32)
- Standard binary (M5A & ASM32)
- Non Standard binary
- Link edited binary (M4EDIT & LNKT32)
- TOC directories (TOC & TOC32)
- TOC save (TOC & TOC32)
- TOC get (TOC & TOC32)

The directive stops listing when the first end of file is encountered.

The $LFILE Directive accepts a search pattern as its last arguments. The contents of the specified file are searched for the supplied pattern and all matched occurences are listed. This feature is supported for directory files (FM, SED, TOC, etc.) as well as data files, however only the name fields of directories are compared against the search pattern. Two special characters (metacharacters) have been reserved to allow expression of more general patterns. The metacharacters "*" matches zero or more occurences of any character and "+" matches one occurance of any character. For example, the search pattern "FM*A" would match "FMA", "FMXA", "FMXXA", etc. The pattern "FM+A" would match "FMXA", "FMBA", etc but not "FMA" or "FMXXA". A restriction exists on directory files that the first and last characters of the entry name to be searched for must be specified either as the exact characters or through metacharacters.

EXAMPLE

To find "abcde" the pattern would need to be something similar to:

"a*cd*" or "a*e"

Embedded blanks are allowed and for this reason, the search pattern must be the last argument(s) specified on this directive.

$LFILE

SYNTAX

	1	2	3	4	5
$LF[[LE]	[logfile]	[FDx] [filename]	[BS[D]] [descriptors]...	[pattern]... [options]...	[pattern]...

[logfile] - Parameter 1 (always defaults specifies a 1- to 3-byte logical file name where the listing output will be sent. Whether it is specified or not, it will always default to the logical filename LO. The device to which this logical filename is assigned can be either a printing or non-printing device (disc, magnetic tape). If assigned to a non-printing device, carriage control is not appended to the output.

[FDx] - Parameter 2 (optional) specifies the File Manager descriptor list FDx. Refer to the MAX IV or MAX 32 FILE MANAGER File Management Manual for further details.

[BS[D]] - Parameter 3 (optional) specifies that the service be performed without the use of the SYSGENed default file descriptor set.

[pattern]... - Parameter 4 (optional) specifies the pattern to be searched for in the contents of the requested file. All matched occurences of this pattern will be listed. The metacharacter "*" matches zero or more occurrences of any character and "+" matches one occurrence of any character. Embedded spaces are allowed.

The alternate form of the directive:

[logfile] - Parameter 1 (optional) is the same as described above for the other format.

[filename] - Parameter 2 (optional) specifies the filename of the file to be listed and has the same format as the filename specified in a FNA descriptor. This parameter can also contain a volumename specification. The volume name is specified by an exclamation (!) followed by the up to 6 character volume name. This is followed by a separator (Example: /) and the full hierarchic filename. Example: !SYSVOL/MYDIR.

[descriptors]... - Parameter 3 (optional) specifies necessary File Manager descriptors instead of specifying the list with separate $FDx directives.

[options]... - Parameter 4 (optional) specifies any of the allowable options defined above for this service (Example: BS[D]). All options to be specified must appear after any descriptors specified in parameter 3 above.

[pattern]... • Parameter 3 (optional) specifies the pattern to be searched for in the contents of the requested file as described in parameter 4 of the first format.

EXAMPLES

$LFILE,,FD0

The File Manager file specified by the file descriptors contained in FD0 is listed to logical file LO.

$LFILE,,MYVOL/MYDIR TNA L00 USR:*

The File Manager file MYDIR on transport L00, volume name MYVOL is searched for occurences of the pattern "USR:*". All matches are listed on logical file LO. Examples of possible matches: "USR:", "USR:A", "USR:LJD", "USR.JDOE".

$LOCATE

LOAD AN OBJECT PROGRAM OR AN INCOMPLETE OBJECT PROGRAM INTO MEMORY AND AT A SPECIFIED MEMORY LOCATION

The $LOCATE Directive loads an object program or incomplete object program into memory as quickly as possible and at a specified memory location. The parameter address specifies the start of memory where the user's program is to be relocated. In MAX IV, a program cataloged in quick-load format is not normally relocated unless it is cataloged PECULIAR RELOCATABLE.

In a MAX IV system, if the address parameter is not entered, the LOCATE address is reset to the start of the host batch task. The address is cleared each time Job Control is loaded and by each $JOB Directive.

SYNTAX

$LOC[ATE] [address]

[address] - Parameter 1 optionally specifies the address of a memory location within the task body.

EXAMPLES

$LOCATE,#8000

An object program is loaded at memory location #8000.

$LOC,#8

An object program is loaded at memory location #8.

$LOC,14000

An object program is loaded at memory location 14000.

REQUEST THE SYSTEM OPERATOR TO MOUNT A VOLUME OR TO VERIFY THAT A REQUIRED VOLUME IS MOUNTED

The $MOUNT Directive is used to request the system operator to mount a volume of media or to verify that a required volume is mounted. The mount request results in a message being printed on the operator's terminal and on the Listing Output device. The service must be completed before any subsequent service invocation causes the volume to be accessed.

SYNTAX

	1	2	3	4	5
$MOU[NT]	[,lfn]	[FDx] [volumename]	[RE[TAIN]] [descriptors]...	[IM[PATIENT]] [options]...	[BS[D]]

[lfn] - Parameter 1 (optional) specifies a 1- to 3-byte logical file name belonging to or being added to the host batch processing task and associated with a File Manager file. This parameter is required if the RETAIN parameter (3) is specified.

[FDx] - Parameter 2 (optional) specifies the File Manager descriptor list FDx. Refer to the MAX IV or MAX 32 FILE MANAGER File Management Manual for a description of what descriptors are applicable to this service.

[RE[TAIN]] - Parameter 3 (optional) specifies that the File Control Block (FCB) associated with the logical file is to be kept. If not specified, the FCB is available for other files.

[IM[PATIENT]] - Parameter 4 (optional) specifies that Job Control does not wait for the volume to be mounted. If not specified, control is not returned until the volume is mounted.

[BS[D]] - Parameter 5 (optional) specifies that the File Manager service should be performed without using the SYSGEN defined default file descriptor list.

The alternate form of the directive:

[lfn] - Parameter 1 (optional) is the same as described above for the other format.

[volumename] - Parameter 2 (optional) specifies the volumename of the volume to be mounted. This form of the command assumes that the parameter is the volume name, the VNA descriptor keyword should not be specified. The detailed description of the format of the volumename is contained in the FILE MANAGER File Management Manual.

[descriptors]... - Parameter 3 (optional) specifies necessary File Manager descriptors instead of specifying the list with separate $FDx directives.

[options]... - Parameter 4 (optional) specifies any of the allowable options defined above for this service (Example: BS[D]). All options to be specified must appear after any descriptors specified in parameter 3 above.

EXAMPLES

```
$FD1 TRANSP=M1 VOLNAM=RAWDATAG
$MOUNT,,FD1
```

A request is made of the system operator to mount volume RAWDATAG on transport M1.

```
$FD6 TNA=F0 VNA=MYSOURCE VOW=RLEE
$MOUNT,,FD6
```

A request is made of the system operator to mount volume MYSOURCE owned by RLEE on transport F0.

```
$MOUNT,,RAWDATAG TNA=M1
```

A request is made of the system operator to mount volume RAWDATAG on transport M1.

REPLACE THE LOGICAL FILE ASSIGNMENTS AND FPI OF THE NEW LOGICAL FILE WITH THE CURRENT LOGICAL FILE ASSIGNMENTS AND FPI OF THE OLD LOGICAL FILE

The $MOVE Directive replaces the logical file assignments and File Position Index (FPI) of the new logical file with the current logical file assignments and FPI of the old logical file. Therefore, the old logical file can be reassigned and used and then restored for further use. The old logical file must be an existing file, but the new logical file is created if necessary and if there is sufficient space in the transient file table in the task.

SYNTAX

	1	2
$MOV[E]	[oldlogfile]	[newlogfile]

[oldlogfile] - Parameter 1 (optional) specifies an already existing logical file.

[newlogfile] - Parameter 2 (optional) specifies a logical file to be given the same logical file assignments and FPI as the old logical file.

EXAMPLES

$MOVE X=SI

The new logical file SI is given the same logical file assignments and FPI as the old logical file X.

$MOV BI SI

The new logical file SI is given the same logical file assignments and FPI as the old logical file BI.

$MSG

SEND A MESSAGE TO A SPECIFIED PORT

The $MSG directive (a JM overlay) is used to send messages preceeded by the Date/Time stamp (if the DTS option is specified in the SYSGEN), to users on a multi-batch system. The intended receiver is identified in one of two ways, either by the terminal (port) number or by the user's system identification name. If the receiver is specified by name, all terminals onto which that user is logged on will receive the message. $MSG uses the PORTINFO REX service to access the User ID information, which is maintained by the MAGIC product.

SYNTAX

$MSG	1 port#	2 message

port# - Parameter 1 (required) is either a port number determined by a terminal's position in a Terminal Control List (TCL), or the intended receiver's User ID (name).

message - Parameter 2 is an ASCII string.

EXAMPLE

$MSG 6 GOOD MORNING!

$MSG JANET GOOD MORNING!

SET A NULL DIRECTIVE

The $NOP Directive is a dummy (null) directive and is ignored by Job Control. It can be used in a job stream or in a procedure to insert a user's comment. The next directive processed is the directive following the $NOP Directive.

SYNTAX

$NOP	[1 text]
[text]	Parameter 1 (optional) specifies a user's comment.

EXAMPLES

$NOP

Job Control processes the next directive after the $NOP Directive.

$NOP THIS IS A SPECIAL PROCEDURE

The comment record $NOP THIS IS A SPECIAL PROCEDURE exists within the procedure. Job Control processes the next directive after the $NOP Directive.

$NOTE

WRITE A MESSAGE TO THE LOGICAL FILE CO WHILE A JOB STREAM IS EXECUTING

The $NOTE Directive writes a message directed to the operator's attention to the logical file CO while a job stream is executing. Job Control is not placed in a HOLD state.

Although only the first three characters (NOT) are required, it is recommended that all four characters be typed to avoid confusion. Refer to the second example below.

SYNTAX

$NOT[E] [1 [text]]

[text] - Parameter 1 (optional) specifies a message directed to the operator's attention.

EXAMPLES

$NOTE PROGRAMS ARE NOW BEING ASSEMBLED
The message $NOTE PROGRAMS ARE NOW BEING ASSEMBLED is written to the logical file CO.

$NOT END OF JOB STREAM
The message $NOT END OF JOB STREAM is written to the logical file CO.

PERFORM ARITHMETIC OPERATIONS ON NUMBERS

The $NUM or $N D rect ve accepts for calculat on, an expression for left to right evaluation with no operator precedence us.ng 32-bit ntegers. If a 16-bit quantity is entered with the sign bit set, the sign s extended before calculations are performed. The result .s printed out in 8-character hexadec mal, 9-digit decimal, and 6-character CAN-code.

SYNTAX

	1	2
$N[UM]	number	[,op,number] ...

number - Parameter 1 (required) can be in one of the following forms:

@ 1-to-6 CAN-code characters excluding blanks

[+] Integer

nteger .s up to an 8-character hexadecima. (preceded by #) or decimal string.

operator (op) - Parameter 2 (optional) can be one of the following:

+ which .ndicates ADDITION
- which .ndicates SUBTRACTION
* which .ndicates MUL TIPLICATION
/ which .ndicates DIVISION

The divide operation dumps the hex remainder and quot.ent. The multiply operation dumps the 64-bit product if 32-bit representation is exceeded, and the result is truncated to 32-bits.

EXAMPLES

```
$NUM #FE*256+5

        0000 FE05    65029
```

OUTPUT

```
$NUM xxxx

  #hhhh hhhh ddddddddd @ccc ccc  'aaaa'
```

Where:

xxxx - item to be converted.

hhhh - 8-character hexadecimal representation.

dddd - 9-character decimal representation.

ccc - 6-character CAN representation.

aaaa - 4-character ASCII representation.

If a division is performed:

```
NUM (FECD) rrrr rrrr qqqq qqqq
```

Where:

rrrr - remainder

qqqq - quotient

```
$NUM #12345678
#1234 5678    305419896   @BOT  M&P  '  &Vv'

$NUM #ABCDEF
#00AB CDEF     11259375   @ DK  5:$  '   '

$NUM @ABCDEF
#0693 19CE    1▲0303694   @ABC  DEF  '   '

$NUM #FE*256+5
#0000 FE05       65029   @          '   '

$NUM #FE00+5
#FFFF FE05        -507   @          '   '

$NUM 15,#2
NUM(FECD) 0000 0001 0000 0007 /OC
#0000  0007          7 @      G
```

MAKE A FILE MANAGER FILE ACCESSIBLE TO THE HOST BATCH PROCESSING TASK

The $OPEN Direct ve makes a F .le Manager f .le accessib.e to the host batch processing task.

SYNTAX

	1	2	3	4	5	6	7
$OPE[N]	logfile	[FDx] [filename]	[IM[PATIENT]] [descriptors]...	[UC[R]] [options]...	[AE[R]]	[BS[D]]	[CR[EATE]] ;

logfile - Parameter 1 (required) spec fies a 1- to 3- character .ogical file name to be assigned to the File Manager file be.ng opened

[FDx] - Parameter 2 (optional) specif es the F.le Manager descriptor list FDx. Refer to the MAX IV or MAX 32 FILE MANAGER File Management Manual for a descript.on of what descr.ptors are applicab.e to this service.

[IM[PATIENT]] - Parameter 3 (optional) specifies that Job Control does not wait if the request cannot be satisfied immediately. If not specif ed, contro. .s not returned unti. the file .s opened.

[UC[R]] - Parameter 4 (optional) specifies that control be returned to Job Control with an error code if usage of the fi e cannot be immediately obtained. If not specif.ed, control is not returned unt.. the desired usage .s obta ned.

[AE[R]] - Parameter 5 (optional) specifies the file should auto expand when reaching the end of the file's current size on read requests as wel. as writes. The f.le must have been created or opened with the AE.I descriptor to use th.s option.

[BS[D]] - Parameter 6 (optional) specif es that the F.le Manager serv ce shou d be performed without us ng the SYSGEN def ned default fi.e descriptor .ist.

[CR[EATE]] - Parameter 7 (optional) specifies that if the F.le Manager returns the error FILE NOT ON VOLUME on the call to OPEN, Job Control calls the CREATE service to create the specified f.le using the descriptor values and options supplied to the OPEN. If this CREATE call completes successfu.ly, Job Control makes a second attempt to OPEN the specif.ed f .e.

The alternate form of the directive:

.ogfi.e Parameter 1 (required) .s the same as described above for the other format.

[f lename] - Parameter 2 (optional) spec fies the filename of the file to be opened. This form of the command assumes that this parameter is

the file name, the FNA descriptor keyword should not be specified. The detailed description of the format of the filename is contained in the FILE MANAGER File Management Manual. This parameter can also contain a volume name specification in the form of: "[volnam]/filename".

[descriptors]... - Parameter 3 (optional) specifies necessary File Manager descriptors instead of specifying the list with separate $FDx directives.

[options]... - Parameter 4 (optional) specifies any of the allowable options defined above for this service (Example: BS[D]). All options to be specified must appear after any descriptors specified in parameter 3 above.

EXAMPLES

$OPEN SCA FD1

The File Manager file is opened and associated with logical file SCA using the file descriptors in the descriptor list FD1.

$OPEN AAA MYFILE

The File Manager file MYFILE is opened and associated with logical file AAA. Note that since no descriptor list was specified, other descriptors necessary to open the file must be defined in the SYSGEN default FDL.

OPEN XXX FILEX TNA T1 VNA V1 ISA 2-T AE1 1-T CR

The File Manager file FILEX is opened and associated with logical file XXX. If the File Manager returns the error FILE NOT ON VOLUME to Job Control on the call to OPEN, Job Control calls the CREATE service using the supplied descriptors. If this CREATE completes successfully, Job Control performs another call to OPEN.

SET SYSTEM AND USER OPTIONS

The $OPTION Directive is an alternative to the options parameter (3) of the $EXECUTE Directive. If the $OPTION Directive is used in addition to placing options in the $EXECUTE Directive, the processing of options begins with the options that are specified in $OPTION Directives, and proceeds with the options that are specified in the $EXECUTE Directive. Refer to the description of the $EXECUTE Directive for more explanation. Table 3-1 lists system options available to the task in the task option word.

Name					Bit in Task Option Word	Purpose if ON (1)
U0	or	0			0	
U1	or	1			1	
U2	or	2			2	Used by various processors and FORTRAN run-time package
U3	or	3			3	
U4	or	4			4	
U5	or	5			5	
U6	or	6			6	Use D26 card conversion
LO	or	U7	or	7	7	Permit Listing Output
BO	or	U8	or	8	8	Permit Binary Output
SC[RATCH]	or	U9	or	9	9	Use SCRATCH file
HO[LD]	or	UA	or	A	10	Enter HOLD state at "various" breakpoints
MA[P]	or	UB	or	B	11	Permit MAP to be output
CO	or	UC	or	C	12	Continue even if errors exist
AO	or	UD	or	D	13	Use Alternative File for commands
HL[OAD]	or	UE	or	E	14	Place task in HOLD state at completion of operation
DU[MP]	or	UF	or	F	15	Dump body of task to global DO file if task should abort

Table 3-1. Task's System Options

$OPTION

SYNTAX

$OPT[ION] [L[INK]]
[[NO] system-options]
[[NO] $user-options]

[L[INK]]
[[NO] system-options]
[[NO] $user-options]

- Parameter 1 (optional) specifies one or all of the following:

LINK causes the system Link Loader to load the program specified in the next $EXECUTE Directive.

system-options specify desired settings of system options (individual bits) in the task option word (for example, MAP, LO or BO). The prefix NO can be used with these options. Valid system option names appear in Table 3-1.

$user-options specify the name of a user option. The first two bytes of this name is passed to the executing program in a register. A maximum of eight user options can be passed. The prefix NO can be used with these options. Refer to the $EXECUTE Directive for more information on the registers used for user options.

EXAMPLES

$OPTION NOLO,NOSC,2,3

The system options NOLO, NOSC, 2 and 3 are set in the task option word.

$OPT HOLD,$CARRY

The system option HOLD is set in the task option word and the user option $CARRY is passed to the next executing program.

SET/RESET SPECIFIED BITS IN THE PROGRAM OPTION WORD

The $POPTION Directive sets/resets the specified bits in the program option word in the TCB. If NO prefixes the program option name, the corresponding program option bit is reset in the program option word. If NO does not prefix the program option name, the corresponding program option bit is set in the program option word.

Name			Bit in Program Option Word
P0	or	0	0
P1		1	1
P2		2	2
.			.
.			.
.			.
P9		9	9
PA		A	10
PB		B	11
PC		C	12
PD		D	13
PE		E	14
PF		F	15

Table 3-2. Task's Program Options

SYNTAX

$POP[TION] [[NO] program option[,...]]

[[NO] program option]

- Parameter 1 (optional) specifies the desired setting of the individual program option bits in the task's program option word. Refer to table 3-2 for valid option names.

EXAMPLES

$POPTION P1,NOP2

The program option bit P1 is set and the bit P2 is reset.

POSITION TO A SED CATALOGED FILE

The $POSITION Directive positions to a SED cataloged file on disc or on a directorized magnetic tape created by a COPY ALL Directive. An error message is generated if the file cannot be found or if there is no directory.

SYNTAX

	1	2
$POS[ITION]	filename	[logfile]

filename - Parameter 1 (required) specifies a 1- to 8-byte name of the file to which the device is to be positioned.

[logfile] - Parameter 2 (optional) specifies the name of a logical file. If not specified, the default value is SI.

EXAMPLES

$POSITION IVIOS

The logical file SI is positioned to the file IVIOS.

$POS SYSGENC USL

The logical file USL is positioned to the file SYSGENC.

DEFINE THE START OF A PROCEDURE

The $PROCEDURE or the $PRODEFAULT Directive defines the start of each new procedure. If more than one of these directives is encountered by Job Control when it is expanding the procedure, the second occurence is assumed to represent the end of the current procedure and the beginning of the next procedure. The file-mark record that defines the end of the log cal file JC is assumed to be the end of the last procedure on that file. The presence of the default parameters on the $PROD Directive definition line does not affect the percent parameter %0. The %0 percent parameter contains the count-of-parameters.

No spaces are allowed between the $ and the directive keyword; however, the $ can be in any column.

SYNTAX 1

```
                    I
$PROC[EDURE]    name-of-procedure
```

name-of-procedure

- Parameter 1 (required) specifies the unique name of the procedure. Up to six characters of the name are used during expansion of the procedure. Therefore, do not specify two procedure names with the first six characters being identical.

SYNTAX 2

```
                          I                    2
$PROD[EFAULT]    name-of-procedure    [parameter default-value]...
```

name-of-procedure

- Parameter 1 (required) specifies the unique name of the procedure. Up to six characters of the name are used during expansion of the procedure. Therefore, do not specify two procedure names with the first six characters being identical.

[parameter-default-value]

- Parameter 2 (optional) specifies the value(s) of one or more default parameters to replace percent parameters during expansion of the procedure. The default values are used when the corresponding parameter is either not supplied on the procedure call line ($DO) or is supplied as an embedded omitted item (,,) on the call line. Refer to the $IFPRESENT and $IFMISSING Directives for a detailed explanation of the exact manner in which percent parameters are replaced.

$PROCEDURE
$PRODEFAULT
$DO PROCEDURE ONLY

In addition a previous percent parameter can be used as a default value, for example:

$PROD NEWPROC,TY,NOLO,%1
%1 Default = TY
%2 Default = NOLO
%3 Default = whatever %1 is therefore in the case where %1 and %3 are missing both %1 and %3 Defaults = TY

Illegal:

$PROD NEWPROC,TY,%4,NOMAP,NOLO
This defaults to an earlier parameter only.

EXAMPLES

$PROC SQUEEZE

A procedure named SQUEEZE is defined. No default values exist for any percent parameters which might appear within the procedure.

$PROD SQUEEZE JSL

A procedure named SQUEEZE is defined. The default value USL will replace the percent parameter %1 if a value is not supplied on the procedure call line.

MODIFY THE FILE MANAGEMENT INFORMATION OF AN EXISTING FILE MANAGER FILE

The $REFILE Directive modifies the File Management Information of an existing File Manager file or the volume specified. The File Manager file must have been previously closed before this directive can be processed.

SYNTAX

	1	2	3	4	5
$REF[ILE]	[logfile]	[FDx] [filename]	[RE[TAIN]] [descriptors]...	[IM[PATIENT]] [options]...	[BS[D]]

[logfile] - Parameter 1 (optional) specifies a 1- to 3-byte logical file name belonging to or being added to the host batch processing task and associated with a File Manager file. This parameter is required if the RETAIN parameter (3) is specified.

[FDx] - Parameter 2 (optional) specifies the File Manager descriptor list FDx. Refer to the MAX IV or MAX 32 FILE MANAGER File Management Manual for a description of what descriptors are applicable to this service.

[RE[TAIN]] - Parameter 3 (optional) specifies that the File Control Block (FCB) associated with the logical file is to be kept. If not specified, the FCB is available for other files.

[IM[PATIENT]] - Parameter 4 (optional) specifies that Job Control does not wait if the request cannot be satisfied immediately. If not specified, control is not returned until the file has been refiled.

[BS[D]] Parameter 5 (optional). The File Manager REFILE service does not use the SYSGEN default FDL except for the transport and volume name (if not specified in caller's FDL). Specifying this option indicates that the default volume and transport should not be used either.

The alternate form of the directive:

[logfile] - Parameter 1 (optional) is the same as described above for the other format.

[filename] Parameter 2 (optional) specifies the filename of the file to be modified. This form of the command assumes that this parameter is the file name, the FNA descriptor keyword should not be specified. The detailed description of the format of the filename is contained in the FILE MANAGER File Management Manual. This parameter can also contain a volume name specification in the form of: "volnam/filename"

$REFILE

[descriptors]... - Parameter 3 (optional) specifies necessary File Manager descriptors instead of specifying the list with separate $FDx directives.

[options]... - Parameter 4 optional, specifies any of the allowable options defined above for this service (Example: BS[D]). All options to be specified must appear after any descriptors specified in parameter 3 above.

EXAMPLE

```
$FD1 FNA=OLDNAME FOW=J.STEVENS
$FD1 FZN=NEWNAME
$REFILE 50 FD1
```

The File Management information for the File Manager file associated with logical file 50 is modified using the file descriptors in the descriptor list FD1.

MODIFY THE VOLUME MANAGEMENT INFORMATION

The $RELABEL Directive modifies the Volume Management Information using the supplied descriptors.

SYNTAX

	1	2	3	4	5
$REL[ABEL]	[logfile]	[FDx] [volumename]	[RE[TAIN]] [descriptors]...	[IM[PATIENT]] [options]...	[BS[D]]

[logfile] - Parameter 1 (optional) specifies a 1 to 3-byte logical file name belonging to or being added to the host batch processing task and associated with a File Manager file. This parameter is required if the RETAIN parameter (3) is specified.

[FDx] - Parameter 2 (optional) specifies the File Manager descriptor list FDx. Refer to the MAX IV or MAX 32 FILE MANAGER File Management Manual for a description of what descriptors are applicable to this service

[RE[TAIN]] - Parameter 3 (optional) specifies that the File Control Block (FCB) associated with the logical file is to be kept. If not specified, the FCB is available for other files.

[IM[PATIENT]] - Parameter 4 (optional) specifies that Job Control does not wait if the request cannot be satisfied immediately. If not specified, control is not returned until the volume has been relabeled.

[BS[D]] - Parameter 5 (optional). The File Manager RELABEL service does not use the SYSGEN default FDL except for the transport and volume name (if not specified in caller's FDL). Specifying this option indicates that the default volume and transport should not be used either.

The alternate form of the directive:

[logfile] - Parameter 1 (optional) is the same as described above for the other format.

[volumename] - Parameter 2 (optional) specifies the volumename of the volume to be relabeled. This form of the command assumes that the parameter is the volume name, the VNA descriptor keyword should not be specified. The detailed description of the format of the volumename is contained in the FILE MANAGER File Management Manual.

$RELABEL

[descriptors]... - Parameter 3 (optional) specifies necessary File Manager descriptors instead of specifying the list with separate $FDx directives.

[options]... - Parameter 4 (optional) specifies any of the allowable options defined above for this service (Example: BS[D]). All options to be specified must appear after any descriptors specified in parameter 3 above.

EXAMPLE

```
$FD1 VOL2NAME=DATAB VOL2OWNER=XXXXX VOLDATE=07/15/82
$RELABEL,,FD1
```

The Volume Management Information for the volume is modified according to the file descriptors in the descriptor list FD1.

POSITION A PHYSICAL DEVICE(S) TO THE BEGINNING-OF-MEDIUM POSITION

The $REWIND Directive positions a physical device or devices to the beginning-of-medium position. If the device(s) is not capable of being rewound, the directive is ignored or some appropriate normalization function is be performed on the device(s). The File Position Index in the File Assign Table is reset to zero.

SYNTAX

$REW[IND] [1] logfile [2] [logfile]...

logfile - Parameter 1 (required) specifies the name of the logical file to be rewound.

[logfile]... - Parameter 2 (optional) specifies additional logical files to be rewound.

EXAMPLES

$REWIND BI,SO

The physical devices associated with logical files BI and SO are rewound.

$REW SI

The physical device associated with logical file SI is rewound.

$SET

SET BACKGROUND MEMORY LOCATIONS BEYOND JOB CONTROL

The $SET Directive sets all background memory locations beyond Job Control to the value specified. If the value is not specified, zero is assumed.

SYNTAX

```
                  1
$SET              [value]
```

[value] - Parameter 1 (optional) is an integer which specifies the value to which memory is set. If not specified, the default value is zero.

EXAMPLES

$SET,#FFFF

All background memory locations beyond Job Control are set to #FFFF.

$SET 1

All background memory locations beyond Job Control are set to 1.

$SET

All background memory locations beyond Job Control are set to zero.

DEFINE A LABEL IN A PROCEDURE

The $TAG Directive defines a label in a directive that can be accessed through a $COUNT or $GOTO Directive. Only the first eight characters of the label name are significant.

SYNTAX

	1	2
$TAG	label-name	[,label-name-2]...

label-name - Parameter 1 (required) specifies a label name consisting of CAN-codeable characters.

[label-name-2]... - Parameter 2 (optional) specifies another label name.

EXAMPLES

```
$TAG XYZ
```

The tag XYZ is defined.

```
$TAG ABC DEF
```

Tags ABC and DEF are defined.

$TAKE

TAKE EXCLUSIVE USE OF DEVICE ASSIGNED TO LOGICAL FILE

The TAKE Directive permits the user to make one or more devices come under the exclusive control of that user's batch task. Such control remains in effect until specifically removed by the user refer the the $GIVE Directive in this manual). The user must refer to the devices by the names of the logical files to which the device(s) are assigned.

SYNTAX

```
                    1
$TAK[E]        logfilname...
```

logfilname - Parameter 1 (required) must be the name of a logical file of the executing batch task or of the system's Global File Assign Table.

EXAMPLES

$TAKE DO

Take Take exclusive use of the device assigned to the DO file.

$ASS X BSA
$TAKE X
Take exclusive use of device BSA to which the logical filename, X, is assigned.

NOTE: This JM directive is not to be used for File Manager data or directory file types.

PRINT A LIST OF DISC TRANSPORTS

The $TNA Directive (a JMA overlay) lists all disc transports in the system.

SYNTAX

```
$TNA
```

EXAMPLE

```
$TNA
TNA  L0   IS  FMGR  S/T = 64    T/C = 4    #CYL = 200    VNA IS .NONE.
TNA  L1   IS  BIOS  S/T = 64    T/C = 4    #CYL = 200
TNA  L00  IS  FMGR  S/T = 42    T/C = 19   #CYL = 411    VNA IS SYSVOL
```

WRITE A FILE-MARK RECORD ON A LOGICAL FILE

The $WEOF Directive writes a file-mark record on each logical file specified.

For some devices, this is a hardware function. For other devices, a software equivalent exists and is detected by handlers during the reading or positioning operations for standard data formats. Refer to the MAX IV BASIC INPUT OUTPUT SYSTEM System Guide Manual for further information on the actions of specific devices to the $WEOF Directive.

SYNTAX

	1	2
$WEO[F]	logfile	[logfile]...

logfile — Parameter 1 (required) specifies the name of the logical file on which a file-mark record is written.

[logfile]... — Parameter 2 (optional) specifies additional logical files.

EXAMPLES

$WEOF B0 12 S1

File-mark records are written on logical files B0, 12, and S1.

$WEO S1

A file-mark record is written on logical file S1.

$WHO

DISPLAY WHO IS ON THIS MULTI-USER SYSTEM

The $WHO Directive (a JM overlay) displays system User IDs, port numbers and the overlays they are running for all terminals having a User ID. This directive uses the PORTINFO REX service to access the User ID information, which is maintained by the MAGIC product.

SYNTAX

$WHO [1 [port...]]

[port...] - Parameter 1 displays who is using this port. If missing all ports are checked for activity

EXAMPLE

```
$WHO

        3       JANE        SED
        4       BETTY       TOC
        5       BILL        JOBCT
       10       KIM         SED
       11       FELIX       JOBCT

$WHO 3

        3       JANE        SED
```

CHAPTER 4
GUIDE TO THE USE OF JOB CONTROL

4.1 JOB CONTROL PROCEDURES

Job Control procedures are sequences of directives in a file stored in a directoried source library or a sequential file. When a Job Control procedure is called either using the $DO Directive or by entering the procedure name as a directive, Job Control searches the logical files UJC and JC for the procedure. If the procedure is found, the source lines within the procedure are copied (expanded) to the work file JW. Logical file CI is assigned to JW and the directives in the procedure are processed. After a procedure has been processed, logical file CI is reassigned to its assignment prior to the procedure call.

The first line of a Job Control procedure is always a $PROCEDURE or $PRODEFAULT Directive. Any source lines previous to these directives are treated as comment lines and are ignored. A procedure is terminated either by an end-of-file mark or by the start of the next procedure. If the name used to call the procedure matches the name under which the procedure is cataloged, the name specified in the $DO call line must match the name specified in the $PROCEDURE or $PRODEFAULT Directive within the procedure.

Job Control procedures can also be stored sequentially within the source library file $$JOB cataloged on logical files UJC or JC. If the Job Control procedures are organized sequentially, a search is made for a procedure with the name that matches the name specified in the $DO call line. In addition, directorized procedures can be on a JSL-type disc directory or on a directorized magnetic tape created through a COPY ALL Directive in the Source Editor.

Lines within procedure files are sometimes modified before they are written to the logical file JW. This occurs when percent parameters are included within the procedure (refer to Section 4.2) and when one of the following directives is included within the procedure: $IF, $IFNOT, $IFMISSING, $IFPRESENT, $EOF. The $PROCEDURE or $PRODEFAULT Directive is never copied to the logical file JW.

EXAMPLE

The logical file UJC is assigned to a disc partition containing the following records in compressed ASCII.

```
* This is a line of comment
$PROC SQUEEZE
$EXE SED
.SQU
.EXI
$$ (End of file)
```

The command $DO SQUEEZE causes the following directives to be processed:

```
$EXE SED
.SQU
.EXI
```

Job Control then continues to read commands from the logical file CI.

4.2 PERCENT PARAMETERS

Procedures can contain variable elements. Variable elements are specified within procedures as %x where x is either a digit in the range 0 through 9 or an ASCII character in the range A through Z. Each occurrence of %1 within the procedure is replaced by up to eight characters of the first parameter supplied when the procedure is called. %A represents the tenth parameter. Parameters after the tenth are represented by the characters B through Z. Up to 35 percent parameters can be defined within a procedure.

The percent parameter %0 is a special case. It is replaced by the character associated with the last argument on the $DO call line.

EXAMPLE

The procedure SQUEEZE from the previous example is modified as follows:

```
* This procedure will squeeze any specified USL file
$PROC SQUEEZE
$EXE SED
.ASS USL %1
.SQU
.EXI
$$
```

The command $DO SQUEEZE BSL causes the following directives to be processed:

```
$EXE SED
.ASS USL BSL
.SQU
.EXI
```

Default values for percent parameters can be specified by including the values in the $PRODEFAULT Directive following the name of the procedure. If a percent parameter is not specified in the $DO call line, the references to it within the procedure are replaced by the appropriate default value.

EXAMPLE

The procedure SQUEEZE is modified again as follows:

```
* This procedure will squeeze any specified USL file, but
* will default to BSL
$PROD SQUEEZE,BSL
$EXE SED
.ASS USL %1
.SQU
.EXI
```

The command $DO SQUEEZE now causes the following directives to be processed:

```
$EXE SED
ASS USL BSL
SQU
EXI
```

The command $DO SQUEEZE CSL causes the following directives to be processed:

```
$EXE SED
.ASS USL CSL
.SQU
.EXI
```

4.3 FILE MANAGER SERVICES

The File Manager is an optional feature of the MAX IV Operating System which, if required, must be installed at system generation.

4.3.1 INVOKING FILE MANAGER SERVICES

All File Manager services are invoked with Job Control statements in one of two formats:

```
  $servicename [logical file name] [descriptor list name] [options]...
Or
  $servicename [logical filename] filename [descriptors]... [options]...
```

The service name can be chosen from the appropriate list of File Manager services summarized in Section 2.4.

The name of the descriptor list to be used by the service must be in the form FDx, where x specifies the desired File Manager descriptor list FD0 through FD9. The list must have already been initialized or must have had a non-default descriptors specified. For file-related services, the descriptor list is optional if a previously invoked service, such as FILEDEFINE, has already associated all required descriptors with the referenced logical file name.

The logical file name must be specified with the file-related services in which the programmer references and uses the real file through a particular logical file name. This capability permits programs to process many different real files since the simple logical file reference used by the program can be associated with other real files external to the coding of the program. The logical file name used must exist permanently for most batch processing tasks, or vacant logical file list entries must be available within the task's control structures to permit logical file definitions to exist for the duration of a job or job step.

The option parameters affect how the File Manager service is executed. If more than one is required, these options can be specified in any order on the service call line.

The alternate form of File Manager Job Control directives allows specification of the filename or volumename for volume related services) in place of the file descriptor list specification. Because of the way the directives determine that parameter 2 is a filename volumename) rather than an FDx, the user cannot use this alternate form if the desired filename is "FDx". For file related services this filename parameter can also contain a volume name specification of the form [volnam/]filename. This capability is also provided when specifying the filename in a $FDx directive (FNA=[volnam/]filename).

The logical filename for this form is the same as described above.

This alternate form also provides the ability to specify descriptors on the service directive rather than specifying separate $FDx directives. Job Control builds an internal file descriptor list using the descriptors supplied and then calls the requested service. This internal descriptor list is reinitialized each time another directive is called. The format of the descriptor keywords and values are the same as for the $FDx directive.

The option parameters must be specified on the call after any descriptors that have been specified. As with the first form, the options can be specified in any order.

When a File Manager service is called and an error is found in either the calling sequence or the descriptor list, the following error message is printed on logical files LO and CO.

FMGR ERROR #nn xxx - error text

The variable nn represents the hexadecimal error message number, the variable xxx represents the error mnemonic, and error text is an ASCII error message (Example: FMGR ERROR #01 JRE - UNRECOVERABLE READ ERROR).

Job Control then aborts if CI is not assigned to a terminal device. Refer to Appendix B for File Manager error codes.

The $JOB and $EOJ Directives normally close all open File Manager files. If closing of files is not desired, Job Control can be recataloged by using the MAX IV Task/Overlay Cataloger (TOC). When recataloging Job Control, the VARIABLE Directive in TOC is used to set the task variable JC= to 1. This prevents File Manager files being closed by $JOB and $EOJ Directives. If the variable JC= is set to 0 or is not present, File Manager files are closed. Refer to Appendix D.

4.3.2 BUILDING FILE MANAGER DESCRIPTION LISTS

Traditional Job Control directives are only one line (for example, a card) in length, and must contain all the parameters necessary to invoke the service. Most parameters can only be specified at fixed positions within the directive and are thus called "positional" parameters.

For File Manager services, single-line Job Control statements with positional parameters are impractical, since the number of parameters required to invoke some of the File Manager services exceed a single line.

Job Control provides special directives for File Manager services. These directives permit lists of File Manager parameters to be defined with separate statements that are used to invoke the services. Because they are separated from the directive that invokes them, such parameters are called "descriptors" to distinguish them from the parameters on the directive line. The directives permit long lists of descriptors to be incrementally built with one or more statements. Once specified, the lists can be referenced by name with service directives that invoke the desired File Manager services. This technique is similar to the way File Manager services are called explicitly in application programs at the MACRO assembly language level.

Job Control permits up to ten independent lists of descriptors to be constructed for use with File Manager service-invoking directives. These separate file descriptor lists are designated FD0 through FD9. To add one or more descriptor specifications to a particular list, only a Job Control statement that references the list by name is used, for example:

```
$FDx [descriptor specification]...
```

where x can have the value 0 through 9, and each descriptor specification is written in one of two formats described below.

Descriptor specifications are not positional but are prefixed by keywords or are entirely keywords. Descriptors that have values or strings of information associated with them are specified in the following way:

```
$FDx [keyword=valuestring]...
```

where keyword is a unique word identifying the descriptor. Most keywords have a short form (three characters) and a long form (six characters). The value string can be a simple numerical value or a name, or a compound string containing several values and/or names separated by the slash (/) delimiter. When compound descriptor values exist, a non-blank delimiter is necessary. Descriptors have a special delimiter set that include the standard set (blanks, comma, slash, and equal sign) as well as a minus sign (-). Since more than one descriptor can reside on one line and since descriptors have a variable number of arguments, a blank denotes the end of the descriptor and its parameters. To delimit compound descriptor values of a descriptor, any of the above delimiters can be used except a blank.

Job Control permits the volume name to be specified as part of a filename specification. The FNA descriptor has the form FNA=volnam/filename if the volume name is to be specified.

Descriptors that can have only simple states associated with them are assigned unique keywords for each state. Such parameters are specified without the need of a value string, for example:

```
$FDx [keyword]...
```

The use of keywords to specify descriptors permits File Manager descriptors to be specified in any order. A list can be built gradually with a series of Job Control statements that refer uniquely to the list FDx. Additionally, more than one descriptor specification can be included in each statement. If a descriptor in a list is specified more than once, the latest specification is used when a File Manager service is invoked.

If a service requires a particular descriptor and that descriptor is missing from the list when the service is invoked, one of the following occurs:

- A default value for the parameter is used.
- A previously specified value for the parameter is used if a previous invocation of the service (with the same logical file reference) included the parameter.
- Job Control notifies the user that a descriptor value is missing.

Each time the Job Control program is reloaded as an overlay between job steps, the ten available descriptor lists are initialized to an empty state. Thus, referencing a list after Job Control is reloaded causes default descriptor values (values specified at SYSGEN) to be used. If a particular descriptor list must be re-used when setting up File Manager services, the list can be explicitly emptied with the following special form of the descriptor list specification statement:

```
$FDx INI or INITIA[LIZE]
```

New descriptors can then be placed in the list. These descriptors can follow on the same statement line that includes the INITIALIZE keyword. This technique is particularly useful if more than ten lists are needed during the set-up of a job or job step.

A particular descriptor list, once specified, can be used for invoking one or more File Manager services. File Manager service descriptors are described in the MAX IV or MAX 32 FILE MANAGER File Management Manual.

4.4 JOB CONTROL LOADING FUNCTIONS

The MAX IV resident executive services that load overlays and tasks can deal only with the quick-load load module format produced by the Task/Overlay Cataloger. This format is satisfactory for production programs that must be installed permanently into the operating system, but is inconvenient to the programmer who is developing new programs, or who has written a program that can never be executed more than once. In these circumstances, a special Sequential Loader or Link Loader can be invoked by Job Control.

4.4.1 SEQUENTIAL LOADER

The MAX IV Operating System provides facilities for loading the sequential output produced by assemblers, macro-assemblers, and linkage editors if such programs are complete (that is, contain no unsatisfied references in the form of internals, externals, or commons). MAX IV detects a load request that references a file not containing quick-load modules, for example:

```
$EXECUTE MAIN,BI
```

where logical file BI is assigned to a sequentially accessible device containing no quick-load modules. In this case, Job Control loads a special overlay called the Sequential Loader into the high end of the addressing space of the batch task and transfers control to it. This loader overlay then loads the requested object program into the low-end of the task's addressing space and transfers control to it. During this process, the amount of actual memory allocated to the batch processing task is expanded to the maximum allowable for that task.

The space used by the loader can be used by the loaded program until Job Control is reloaded upon program exit. The sequential loader is unprivileged and can only load programs in unprivileged mode.

4.4.2 LINK LOADER

If a program is incomplete (that is, contains unsatisfied references) and if standard libraries of subroutines (for example, logical files LB or SL) are required to complete the program, the specially defined LINK option can be requested on the $EXECUTE Directive. This option causes the special Link Loader overlay to be loaded into the high-end of the addressing space of the batch processing task. This loader operates identically to the Sequential Loader, except that a symbol table is constructed between the loader and the program being linked and loaded from the low-end up. When control is transferred to the loaded program, it can utilize the expanded memory area used for the loader and its symbol table. The extra space is deallocated when Job Control is reloaded after the program exits, and privileged mode is restored.

When the Link Loader facility is used on a MAX IV system, note that it cannot take full account of counters and attributes (caused by the CTR and ATR directives to the Macro Assemblers and present in some MAX IV Language Processor output) or of global or extended common. The Link Loader allows a counter (without attributes) to be declared at the start of a module. Later declaration of a different counter or declaration of any attributes causes an illegal function code error.

The Link Loader can access libraries in sequential format, but cannot access object libraries in directoried format.

4.4.3 OTHER BATCH PROCESSING PROGRAMS

Standard system processor programs are normally installed on the same global file on which Job Control is installed. These system processors normally operate in unprivileged mode, except the Task/Overlay Cataloger in MAX IV. These programs can be installed as privileged overlays if the system programmer wants a particular task to have the capability of installing privileged tasks and overlays into the operating system. Otherwise, these processors can be restricted in their program installation capabilities, so that they can only install other unprivileged programs.

Many standard system processor programs are available and are described in the appropriate programmer's reference manual.

APPENDIX A
JOB CONTROL ERROR MESSAGES

ERROR MESSAGE	REASON
COMMAND ERROR	The syntax of a direct va ls incorrect (for examp.e, it did not start with a $ sign and logical file CI was not assigned to a terminal), or an error has been detected in processing a nonresident directive.
xxxxxxxx NOT PRESENT	The $POS command was used to search for a file within a directory, and the fi.e was not found.
GO ERRORS	An attempt has been made to execute a program from Job Control after a previous program (assembler, compiler, etc.) has set a flag to show that an error has occurred.
ILLEGAL FILE/DEVICE NAME	A logical fi.e name was not CAN-codeable or was not defined.
xxxxxxxx UNDEFINED	A $GOTO or $COUNT Directive resulted in a jump to a label that could not be found within this procedure.
NO PROCEDURE	The procedure specified in a $DO statement was not present.
NO PROGRAM	The program specified in a $EXECUTE statement could not be found on the load module specified.
ILLEGAL TAG	A tag name was not CAN-codeable.
MULTIPLE DEFINED TAG	Two tags were defined with the same name.
SYMBOL TABLE OVERFLOW	More than 25 tags or forward $GOTO Directives were encountered in a procedure.
NO $JOB COMMAND	An error has occurred and a $JOB or $CJOB Directive is required before subsequent directives can be processed.
NO VACANT ENTRIES	An attempt to assign a new logical file to a device (or another logical file) failed. There are no additional vacant FAT tables in the task's resources. The user can attempt to free up some FAT tables by using $JOB or $ASSIGN a logical file to itself.
CANNOT ASSIGN FMFILE	An attempt was made to assign a File Manager FAT. This is NOT legal

APPENDIX B
FILE MANAGER ERROR CODES

When a File Manager service is called and an error is found in either the calling sequence or the descriptor list, the following error message is printed on logical files LO and CO.

FMGR ERROR #nn xxx - error text

The variable nn represents the hexadecimal error message number, the variable xxx represents the error mnemonic and error text is a short ASCII text string describing the error. Job Control then aborts if CI is not assigned to a terminal device.

MNEMONIC	HEX	FORTRAN DECIMAL	MEANING
URE	1	2	Unrecoverable read error
NVE	2	3	No vacant File Assign Table (FAT) in Task Control Block (TCB) or no vacant Logical Device Table (LDT) in system transients
LFN	3	4	File Control Block (FCB) retention was specified but no Logical File Name (LFN) was supplied
SRL	4	5	No space in the TCB's system region list for the FCB
MZU	5	6	Map 0 usage limit exceeded
DIC	6	7	Illegal descriptor in a FDLINCLUSION descriptor or a ENDFDLIST descriptor is missing
RFN	7	8	ROOTNAME descriptor follows FILENAME or FILE2NAME descriptors in FDLINCLUSION descriptor
FLF	8	9	First-level FILENAME length is greater than the SYSGEN maximum
CFN	9	10	Complete FILENAME length is greater than the SYSGEN maximum
NTR	A	11	No TRANSPORT name was specified
TRU	B	12	TRANSPORT name was undefined on the system
VNM	C	13	Volume not mounted; transport not available
UPE	D	14	User permissions do not allow operation
BIO	E	15	Basic I/O system error

MNEMONIC	HEX	FORTRAN DECIMAL	MEANING
CKS	F	16	Checksum or record code error
NSM	10	17	No space is available for management allocation
ECE	11	18	Exceeded expansion limit count
EMS	12	19	Exceeded maximum space
PNI	13	20	Partition file is not an integral number of tracks
TGM	14	21	Total size request is greater than the maximum
TLC	15	22	Total specified is less than contiguity
PNC	16	23	Partition file is not wholly contiguous
IGD	17	24	Contiguity requested is incompatible with allocation pool granularity
DNS	18	25	Required descriptor was not specified
UNL	19	26	User ID was not in list
INL	1A	27	INFLUENCELIMIT descriptor is not compatible
CNI	1B	28	Contiguity is not an integral number of tracks
GEO	1C	29	Geometry incorrect for transport or incompatible with granularity
VND	1D	30	Volume is not dismountable now
FAE	1E	31	FILENAME already exists on the volume
NVN	1F	32	No volume name supplied
FNV	20	33	FILENAME is not on volume
DPI	21	34	Device Position Index (DPI) is not accessible through the map structure
VNL	22	35	Volume is not labeled
VAL	23	36	Volume is already labeled
NSF	24	37	No space is available for file allocation
FNO	25	38	File is not open or should be open
FAO	26	39	File is already open or should not be open

MNEMONIC	HEX	FORTRAN DECIMAL	MEANING
NDF	27	40	Simple FILENAME is not a directory file
DNO	28	41	Cannot do ENDFILE to directory file
PNF	29	42	Partition data file is not a first-level file
IRR	2A	43	An IMPATIENT parameter was specified within a directive and the request was refused
VPM	2B	44	Volume is permanently mounted
RDZ	2C	45	Required descriptor was supplied with a zero value
NOTE	Refer to the MAX IV or MAX 32 BASIC INPUT/OUTPUT SYSTEM System Guide Manual for detailed information on the next seven error conditions.		
DOH	2D	46	Pre-I/O OTH or HO error
NNA	2E	47	No I/O node available
DIO	2F	48	Device inoperable
OTH	30	49	Post-I/O OTH, HO, or SBV error
LOK	31	50	Post-I/O LOK error
DTP	32	51	Post-I/O parity error
DTS	33	52	Post-I/O DATA SYNC OVF/UNF error
CED	34	53	Cannot expand/contract directory files
CEP	35	54	Cannot expand/contract partition data files
CCM	36	55	Cannot contract multiply-opened files
NUL	37	56	No UID/Permlist exists
FNC	38	57	Name of a partition data file is not CAN-codable
SDU	3A	59	Sequential devices are unimplemented
UIL	3B	60	User ID is already in the list
NVM	3C	61	No volume mounted on transport
FNR	3D	62	File Manager is not currently responsible for transport
WVM	3E	63	Wrong volume mounted on transport
WND	3F	64	WDI is not an opened directory

APPENDIX C
EXAMPLE OF A JOB CONTROL OVERLAY

This program is an example of a Job Control overlay.

```
1           PGM     ADD
2           INS*    MC,IVEQUATE
3           INS*    MC,IVUEQU
4   ADD     EQU     $
5           TRR,R1,R14              SAVE "NORMAL" RETURN ADDRESS
6           TRR,R3,R11              SAVE "ERROR" RETURN ADDRESS
7           TRR,R4,R13              SAVE "INVALID PARAM" RETURN ADDRESS
8           TRRD,R10,R6             GET PARAMETER ADDRESS AND OFFSET
9           LDI,R8      COLLECT     COLLECT NEXT PARAMETER SERVICE
10          REX,MAXIV               ISSUE REX CALL
11          HCS,ERR2                INVALID PARAMETER
12          HNR,ERR1                NO PARAMETER
13          LDI,R8      ATN         ASCII TO NUMERIC SERVICE
14          REX,MAXIV               ISSUE REX CALL
15          HNR,ERR2                INVALID NUMBER
16          STMD,R12    NUM1        SAVE FIRST NUMBER
17          TRRD,R6,R10             SAVE START OF SECOND PARAMETER
18          LDI,R8      COLLECT     COLLECT NEXT PARAMETER SERVICE
19          REX,MAXIV               ISSUE REX CALL
20          HCS,ERR2                INVALID PARAMETER
21          HNR,ERR1                NO PARAMETER
22          LDI,R8      ATN         ASCII TO NUMERIC SERVICE
23          REX,MAXIV               ISSUE REX CALL
24          HNR,ERR2                INVALID NUMBER
25          LDMD,R14    NUM1        RETRIEVE FIRST NUMBER
26          ADRD,R12,R14            AND ADD THE TWO TOGETHER
27          LDI,R8      DTD         CONVERT 32 BIT NUMBER TO ASCII
28          LDI,R14     -1          NO DECIMAL POINT WANTED
29          REX,MAXIV               ISSUE REX CALL
30          HNR,ERR1                NUMBER LARGER THAN 9 BYTES
31          SFM,R11     RESULT      SAVE RESULT IN PRINT BUFFER
32          LDI,R8      WRITE       WRITE SERVICE
33          LDI,R2      LOUFT       UFT FOR WRITE
34          LDI,R14     MESS        ADDRESS OF MESSAGE TO PRINT
35          LDI,R15     26          NUMBER OF BYTES TO PRINT
36          REX,MAXIV               ISSUE REX CALL
37          BRX,R1                  NORMAL RETURN
38  ERR1    EQU     $
39          BRX,R3                  ERROR EXIT
40  ERR2    EQU     $
41          BRX,R4                  INVALID PARAMETER ERROR
42  *
43  NUM1    RES     2
44  MESS    DFC     'THE RESULT IS *
45  RESULT  RES     5,0
46  LOUFT   DFC     0,@LO,#A000,0,0,0,#4000,0,0,0
47          END     ADD
```

APPENDIX D
INSTALLATION CONSIDERATIONS

This appendix contains installation considerations in the following areas:

- The assembly options for Job Control
- The File Manager file close option through TOC
- An installation procedure

D.1 ASSEMBLY OPTIONS FOR JOB CONTROL

When the Job Control system processor is assembled from the source code, the following options are available:

Option	Meaning
$C1	The $ prompt is output to the logical file C1 rather than the logical file CO.
$JO	The $JOB or the $CJOB Directive is required after a program abort.
$NC	Open File Manager files are not closed by a $JOB or $EOJ Directive. This option can also be specified through recataloging Job Control through TOC. Refer to Section D.2.

D.2 FILE MANAGER FILE CLOSE OPTION THROUGH TOC

As the default condition, the $JOB and the $EOJ Directives close all open File Manager files. Job Control can be recataloged using the VARIABLE Directive in the MAX IV TASK/OVERLAY CATALOGER (TOC). The task variable JC: is used to control this option.

If JC: is set to 1, open File Manager files are NOT closed after a $JOB or $EOJ Directive. If JC: is set to zero or is not specified at all, open File Manager files are closed. The following workflow of directives recatalogs the standard version of Job Control as the no-close version.

```
$JOB
$EXE TOC
FIL BM
ASS BI SO
GET B
$OU B INIT
VAR JC: 1
PUT,,NEW
EXI
```

D.3 INSTALLATION PROCEDURE

The following installation procedure installs Job Control.

```
        TTL     JOB CONTROL INSTALLATION PROCEDURE
THIS PROCEDURE WILL INSTALL JOB CONTROL
PARAMETERS ARE
%1 = LOAD MODULE FILE (DEFAULT BM)
%2 = ID (DEFAULT 08.85H.2)
%3,%4 = ASSEMBLY OPTIONS (DEFAULT NOLO,NOBO)
%5 = NAME USED TO CATALOG JOB CONTROL (DEFAULT B)
$PROD JOBGEN,,08.85H.2,NOLO,NOBO,B
$ASS SI JC UJC NO BI BI BO BO
$REW SI BI BO
$AVF SI
$EXE MSA,,NOSC,%3,%4
$WEO BO
$REW BI
$EXE TOC
$IFP %1 P FILE %1, FILE BM
TASK %5 #FF #7F
PEC AUTOSTART
!OPERATIONS #03 #00
STACKS #0096 #00A0
PAGESHARING #04
SYSPAGES #09
ERASE IMAP ALL
DEALLOCATE IMAP ALL
OPTION NO0
OPTION NO1
OPTION NO2
OPTION NO3
OPTION NO4
OPTION NO5
OPTION NO6
OPTION NOA
OPTION NOC
OPTION NOD
OPTION NOE
OPTION NOF
OPTION 7
OPTION 8
OPTION 9
OPTION B
LOG CI TY
LOG CO TY
LOG LO TY
LOG LS SLS
LOG JW BJW
LOG SCA BSA
```

```
LOG SCB BSB
LOG SC BSC
LOG SI TY
LOG SO BSB
LOG BI BSA
LOG BO BSA
LOG JSL BSL
LOG SCC BSW
LOG AI TY
LOG . TY
LOG J LP
LOGFILES #0A
LOG WRK BSW
LOG FMM MC
LOG FM MC
LOG RAD NO
LOG LB ALB
LOG LM GLM
PEC PRI
PEC UNA
PEC BAT
PEC REMOTE
VARIABLES #03
ISPACE 0,ALL
OSPACE 0,ALL
ID %2
CAT
EXI
$END
```

Job Control must be cataloged with a variable extension to enable the OC Directive RESUME to pass information to system processors.

EXAMPLE

/R MAP The MAP request is passed to the Link Editor system processor.

Refer to the MAX IV TASK/OVERLAY CATALOGER Programmer's Reference Manual, in particular to the VARIABLE Directive.

INDEX

Fold

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 3684 FT. LAUDERDALE, FL 33304

POSTAGE WILL BE PAID BY ADDRESSEE

MODULAR COMPUTER SYSTEMS
1650 W. McNAB ROAD
P.O. BOX 6099
FT. LAUDERDALE, FLORIDA 33310

Attention: TECHNICAL PUBLICATIONS, M.S. #85

Fold

MODCOMP

Please comment on the publication's completeness, accuracy, and readability. We also appreciate any general suggestions you may have to improve this publication.

If you found errors in this publication, please specify the page number or include a copy of the page with your remarks.

Your comments will be promptly investigated and appropriate action will be taken. If you require a written answer, please check the box and include your address below.

☐

Comments: _____

Manual Title _____

Manual Order Number _____ Issue Date _____

Name _____ Position _____

Company _____

Address _____

_____ Telephone ( ) _____

MODCOMP

Corporate Headquarters:
MODULAR COMPUTER SYSTEMS, Inc., 1650 West McNab Road, P.O. Box 6099, Ft. Lauderdale, FL 33310, Tel. (305) 974-1380, TWX: 310-372-7837

International Headquarters:
MODULAR COMPUTER SERVICES, Inc., The Business Centre, Molly Millars Lane, Wokingham, Berkshire, RG11 2QJ, UK, Tel. 0734-796-939, TLX: 851 849149

Latin American Sales Headquarters:
MODULAR COMPUTER SYSTEMS, Inc., 1650 West McNab Road, P.O. Box 6099, Ft. Lauderdale, FL 33310, Tel. (305) 975-6562, TLX: 3727652

Canadian Headquarters:
MODCOMP Canada, Ltd., 400 Matheson Blvd. East, Unit 29, Mississauga, Ontario, Canada L4Z 1N8, Tel. (416) 890-0866, TELEX: 06-961279

SALES & SERVICE LOCATIONS THROUGHOUT THE WORLD

Printed in the U.S.A.